University of Tampere

Department of Information Studies

Research Notes

RN • 1999 • 1

KALERVO JÄRVELIN, PETER INGWERSEN & TIMO NIEMI

INFORMETRICS THROUGH ADVANCED DATA MANAGEMENT:

Complex Object Restructuring, Data Aggregation and Transitive Computation

Tampereen yliopisto • Informaatiotutkimuksen laitos • Tiedotteita
1999 • 1

INFORMETRICS THROUGH ADVANCED DATA MANAGEMENT: Complex Object Restructuring, Data Aggregation and Transitive Computation

Kalervo Järvelin+, Peter Ingwersen* and Timo Niemi#
+Dept. of Information Studies
#Dept. of Computer Science and
University of Tampere
P.O.Box 607
FIN-33101 TAMPERE, Finland

*Royal School of Librarianship
Birketinget 6
DK-2300 COPENHAGEN S, Denmark

INFORMETRICS THROUGH ADVANCED DATA MANAGEMENT

Kalervo Järvelin^{+}, Peter Ingwersen* and Timo Niemi$^{\#}$
$^{+}$Dept. of Information Studies
$^{\#}$Dept. of Computer Science and
University of Tampere
P.O.Box 607
FIN-33101 TAMPERE, Finland

*Royal School of Library and Information Science
Birketinget 6
DK-2300 COPENHAGEN S, Denmark

ABSTRACT

This article considers how informetric calculations can easily and declaratively be specified through advanced data management techniques. In particular, bibliographic data and its modeling as complex objects (non-first normal form relations) as well as terminological and citation networks involving transitive relationships are considered. A very high-level declarative query interface, based on this data model, is introduced. The article demonstrates that such data modeling and query interface enable end-users to perform basic informetric ad hoc calculations, such as bibliographic coupling, author co-citation analysis, generalized impact factors, international visibility and international impact, productivity calculations in a given area, etc., easily and often with much less effort than in the contemporary online retrieval systems. Several fruitful generalizations of typical informetric measurements are also proposed. These are based on substituting traditional foci of analysis, e.g., journals, by other object types, such as authors, organizations, countries or classes of a classification scheme. It is shown that the proposed data modeling and query interface make it trivial to switch focus between various object types for informetric calculations. Moreover, it is demonstrated that all informetric data can easily be broken down by criteria that foster advanced analysis, e.g., by years or content-bearing attributes. Such modeling allows flexible data aggregation along many dimensions and the utilization of transitive relationships. These salient features emanate from the query interface's general data restructuring and aggregation capabilities combined with transitive processing capabilities. The features are illustrated by means of sample queries and results.

1. INTRODUCTION

Informetrics studies various statistical phenomena of literature often based on bibliographic information provided by online databases. Among the statistical phenomena are productivity issues (of authors, countries, journals; Almind & Ingwersen, 1997), generalized impact factors (of journals, authors; Hjortgaard Christensen & al., 1997), activity profiles (of authors, organizations, journals), citation networks (bibliographic coupling of authors or articles, author co-citations; White, 1990), literature growth and aging, to mention a few (Library 1981).

Several informetric measurements are produced by the ISI (Institute of Scientific Information), published in their reports, e.g., the Journal Citation Report. Informetric calculations can also be done online in the online databases. Hjortgaard Christensen and Ingwersen (1996; & Wormell, 1997) have described the methodology of various citation-based analyses using the OneSearch, RANK and TARGET commands of the Dialog Information Service. Very often ad hoc informetric measurements are needed for decision making, e.g., for competitor information, science policy, research project funding, etc.

This article considers how informetric measurements can easily and declaratively be specified through advanced data management techniques. We consider typical informetric data, i.e., bibliographic data as well as terminological and citation networks involving transitive relationships. The poor capability of the conventional relational model in modeling and processing complex objects in many applications, including information retrieval (IR), has led many researchers to study the NF^2 relational model (non-first normal form relations; Sheck & Scholl, 1986), object-oriented databases, deductive databases and deductive object-oriented databases (e.g., Deux, 1990; Ullman, 1989; Paredaens & *al.,* 1995). Bibliographic data are naturally modeled as NF^2 relations (Desai & *al.*, 1987; Niemi & Järvelin, 1995). Moreover, some terminological network structures, e.g., thesauri and classifications, and citation networks cannot be modeled by the NF^2 principle. Thus we shall model such structures as binary relations which support computations involving transitive relationships (Agrawal & Borgida & Jagadish, 1989; Järvelin & Niemi, 1997; Niemi & Järvelin, 1992). The paper demonstrates how the management of NF^2 relations and transitive relationships is integrated.

We shall introduce briefly a very high-level declarative query interface based on NF^2 relations and transitive relationships (Järvelin & Niemi, 1995; 1997). This interface, called the FUN interface, provides general data restructuring and aggregation capabilities combined with general transitive processing capabilities thus providing powerful features for retrieving and analyzing bibliographical and citation data. The need for such capabilities has been recognized necessary in many document and IR related studies (see, e.g., Macleod, 1990; Rada & *al.,* 1993; Salminen & *al.,* 1994). The data aggregation capabilities of online IR systems fall short for several informetric measurements. For example, Dialog's Rank feature (Dialog, 1993) and ESA-IRS's Zoom feature (Ingwersen, 1984) merely provide term counts in a single field of a retrieved set of documents. Persson's recently developed bibliometric toolbox, available on the Net, is limited to productivity data only (1999). We shall show that general data restructuring and multi-level aggregation are necessary for informetrics. In addition to these general capabilities, an essential feature of the FUN interface is its very high abstraction level and declarativity. The user need not specify how the results are derived from the database. Instead, the interface deduces the derivation steps even in complex query situations.

This article demonstrates that the proposed data modeling and query interface enable end-users to perform basic informetric ad hoc calculations, such as bibliographic coupling, author co-citation analysis, generalized impact factors, international visibility and international impact, productivity calculations in a given area, etc., easily and often with much less effort than in contemporary online retrieval systems. For instance, users need not determine in advance a set of all author pairs for co-citation analysis and derive the data separately for each pair — this is done by a single query. We shall also propose several fruitful generalizations of typical informetric measurements. These are based on substituting traditional foci of analysis, e.g., journals, by other object types, such as authors, organizations, countries or classes of a classification scheme. It is shown that the FUN interface makes it trivial to switch focus between various object types for informetric calculations. Moreover, it is demonstrated that all informetric data can easily be broken down along several dimensions that foster advanced analysis, e.g., by years or content-bearing attributes. Thus, our data modeling and query interface support generalized informetrics. As a spin-off effect, citation data may be used for IR purposes. This is an area of IR research which has been neglected in recent years.

Ingwersen and Hjortgaard Christensen (1997) have pointed out that the quality of database contents is essential for informetric analysis. In this paper we shall not consider the problems caused by real bibliographic databases containing corrupted, incomplete data, and partially incompatible data, e.g., varying journal names in citations. Instead, we shall utilize an abstracted bibliographic sample database, not suffering from such problems, for our analyses. In practice, our interface is dependent on the quality of downloaded data. However, this is a problem to be considered also in all other approaches. The quality problems are no worse for our approach than in the traditional online situations. Although the database is realistic it is not real, and thus the sample queries and results we present are only illustrative for our approach.

This paper extends our previous work on our NF^2 relational query interface (e.g., Niemi & Järvelin, 1995), its data aggregation features (Järvelin & Niemi, 1995) and its transitive processing capabilities (Järvelin & Niemi, 1997) to a new application area, informetrics. The contributions for informetrics are both methodological, i.e., related to data modeling and computation of informetric measurements, and conceptual, i.e., providing fruitful generalizations for informetric concepts.

The paper is organized as follows. Section 2 presents our sample database environment and exemplifies the data modeling principles. Section 3 reviews the features of the proposed query interface and summarizes the query language. Section 4 discusses, through sample queries, how data restructuring and aggregation as well as transitive computation are used through our declarative query interface in informetric computation. Comparisons with traditional online informetrics are provided. Generalizations of informetric concepts are also presented. Sections 5 and 6 contain discussion and conclusions.

2. SAMPLE DATABASE ENVIRONMENT

Figures 2.1 - 2.4 exemplify three data modeling situations where two kind of modeling principles are necessary. Figures 2.1 a-b show the data structure diagram and a sample instance of a complex object, or NF^2 relation, representing bibliographic references. In the diagram, rectangles represent relation-valued attributes while ellipses represent the atomic-valued attributes (or properties) of each relation-valued attribute. Thus the complex object ARTICLES has two levels with the relation-valued attribute ARTICLES forming the *top relation*, and the relation-valued attributes AUTHORS, KEYWORDS and CLASSES forming its *immediate subrelations.* The latter relation-valued attributes are *bottom relations.* The sample instance in Fig. 2.1. (b) shows five articles from three different journals, having one or more authors (with affiliations), and several keywords as well as several classes. Complex objects of type ARTICLES are structurally static in the sense that all objects have exactly two levels. No recursive structure is present. Complex objects of type ARTICLES are formed from more simple objects of various types and are naturally represented by NF^2 relations.

The relation ARTICLES has the atomic-valued attributes *ano* (article number), *title* (article title), *publisher* (publisher number), *journal* (journal name), *year* (publication year), *vol* (journal volume), *issue* (journal issue) and the three relation-valued attributes AUTHORS, KEY-

WORDS and CLASSES. These latter three attributes contain the atomic-valued attributes *author* (article author), *department, organization, city and country* (which give the author's affiliation), *key* (a thesaurus term) and *class* (a class of the ACM Computer Science Classification), respectively. NF^2 relations are excellent for modeling structurally static complex objects such as the relation ARTICLES. They are not suited for modeling structurally dynamic objects like thesauri or citation networks which have, in principle, unlimited acyclic transitive relationships between nodes.

In some contemporary public databases, for instance, the citation databases produced by the Institute for Scientific Information (ISI), there is no direct link between each author and his/her affiliation. Such a limitation in the input data must be resolved prior to data analysis for all queries which require the affiliation data per author.

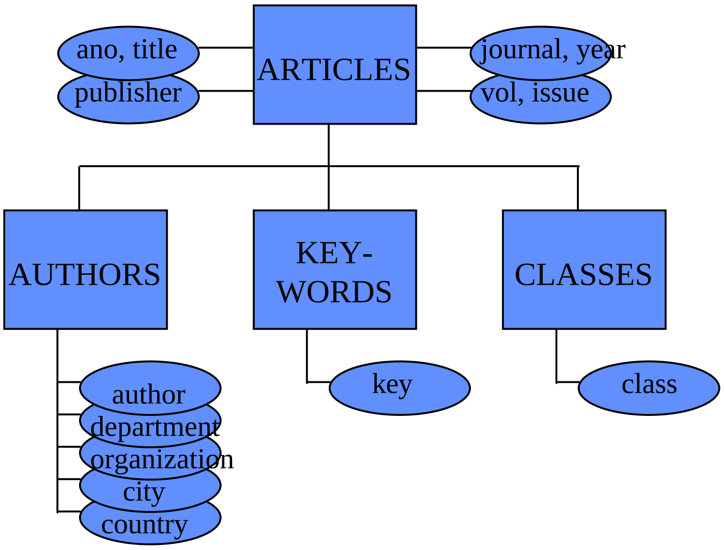


Fig. 2.1 (a) Modeling bibliographic references as complex objects: the data structure diagram

Our query interface employs a linear data structure representation called *form.* A form gives the relation-valued and atomic-valued attribute names of a relation and employs parentheses to denote the nesting level of each component. The form ARTICLES(ano, title, publisher, journal, year, vol, issue, AUTHORS(author, department, organization, city, country), KEYWORDS(key), CLASSES(class)) corresponds to the data structure diagram of Figure 2.1a. We follow the convention of marking relation-valued attribute names in capital letters and atomic-valued attribute names in lower case letters.

```
{(art_1, The relational model in information retr, John Wiley & Sons,
  JASIS, Journal of the American Society f..., 32, 1, 1980,
  {(Crawford, R, Department of Computing ..., Queens Univer..., Kingston, Canada)},
  {(relational database), (sequel), (relational algebra),(bibliographic database)},
  {(H.2.1), (H.3.3)}),

 (art_3, Universal relation theory applied to bib, The Canadian Association for
  Information, The Canadian Journal of Information Scie, 9, 1, 1984,
  {(Crawford, R, Department of Computing ..., Queens University, Kingston, Canada),
   (Becker, S, Department of Computing ..., Queens University, Kingston, Canada),
   (Ogilvie, J, Department of Computing ..., Queens Univer..., Kingston, Canada)},
  {(relational database), (universal relation), (bibliographic database)},
  {(H.2.1), (H.3.3)}),

 (art_5, Non-first normal form universal relation, Pergamon, Information Systems,
  12, 1, 1987,
  {(Desai, B, Department of Computer Sci…, Concordia University, Montreal, Canada),
   (Sadri, F, Department of Computer Sci…, Concordia University, Montreal, Canada),
   (Goyal, P, Department of Computer Sc…, Concordia University, Montreal, Canada)},
  {(non-first normal form relation), (universal relation),…, (document retrieval)},
  {(H.2.3), (H.2.4), (H.3.3)}),

 (art_20, Deductive Information Retrieval Based on, John Wiley & Sons,
  JASIS, Journal of the American Society f..., 44, 10, 1993,
  {(Niemi, T, Department of Computer Sc…, University of Tampere, Tampere, Finland),
   (Jarvelin, K, Department of Inf… St…, University of Tampere, Tampere, Finland)},
  {(query languages), (knowledge-based retrieval), (deductive database), ...},
  {(H.2), (H.3.2), (H.3.3), (I.2.4)}),

 (art_21, Text Retrieval and the Relational Model, John Wiley & Sons,
  JASIS, Journal of the American Society f..., 42, 3, 1991,
  {(Macleod, I, Department of Computing …, Queens University, Kingston, Canada)},
  {(relational database), (text retrieval), (query languages)},
  {(H.2.2), (H.2.1), (H.3.2), H.3.3)}),
... }
```

Fig. 2.1 (b) Modeling bibliographic references as complex objects: partial instance

Figure 2.2 shows the data structure diagram and a sample instance of the thesaurus TERM objects and their thesaural relationships. Each TERM object is atomic, i.e., there is only the Term-Name attribute in each object. The data structure diagram shows that thesaurus terms are related to themselves through the narrower term (SUBTERM) relationship. The relationship SUBTERM is transitive, i.e., if document retrieval is an *immediate subterm* of data management and query formulation is an *immediate subterm* of document retrieval, then query formulation is a *transitive subterm* of data management. The sample instance shows an excerpt of terms in the hierarchic SUBTERM relationship.

Thesaurus objects are structurally dynamic in the sense that they have unlimited acyclic transitive relationships with varying depth in different directions from any given TERM object, i.e., the structure is recursive. Thesaurus-like structures cannot be modeled as structurally static complex objects, like NF^2 relations. They can, however, be modeled through binary relations representing transitive relationships indirectly.

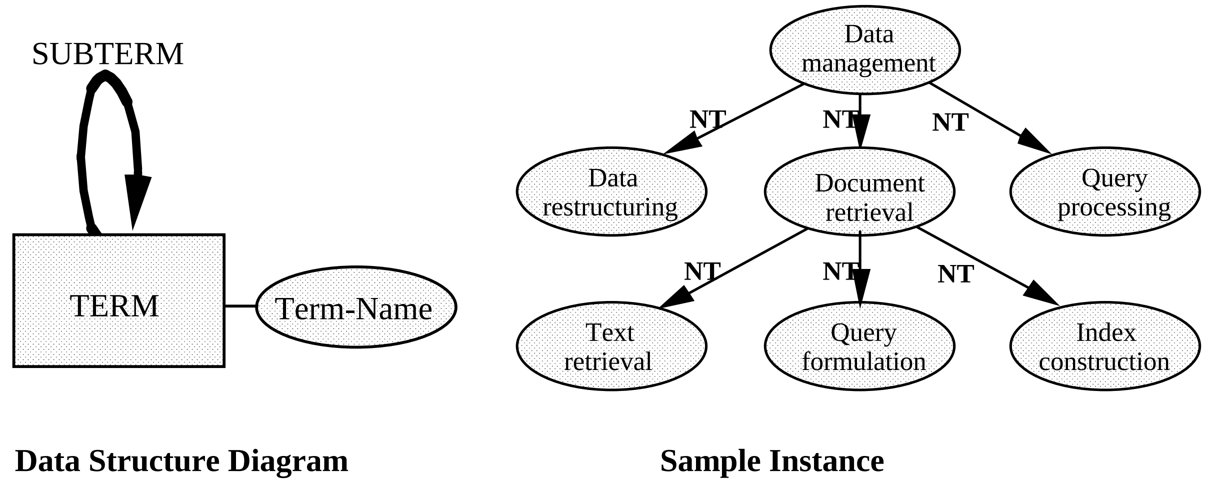


Fig. 2.2. Modeling a transitive hierarchic relationship

Figure 2.3 shows the data structure diagram and a sample instance of simple ARTICLE objects and their citation relationships. Each ARTICLE object is atomic, i.e., there is only the *ano* attribute in each object. The data structure diagram shows that articles are related to themselves through the transitive citation (CITES) relationship. The sample instance shows an excerpt of a citation network. Also citation networks are structurally dynamic in the sense that they have unlimited acyclic transitive relationships with varying depth from any given ARTICLE object. The reference list of an article — the cited articles — could well be represented as a relation-valued attribute of an article in an NF^2 relation. However, this would not support finding *citing articles* of a given article. Thus citation networks are not usefully modeled as structurally static complex objects, like NF^2 relations.

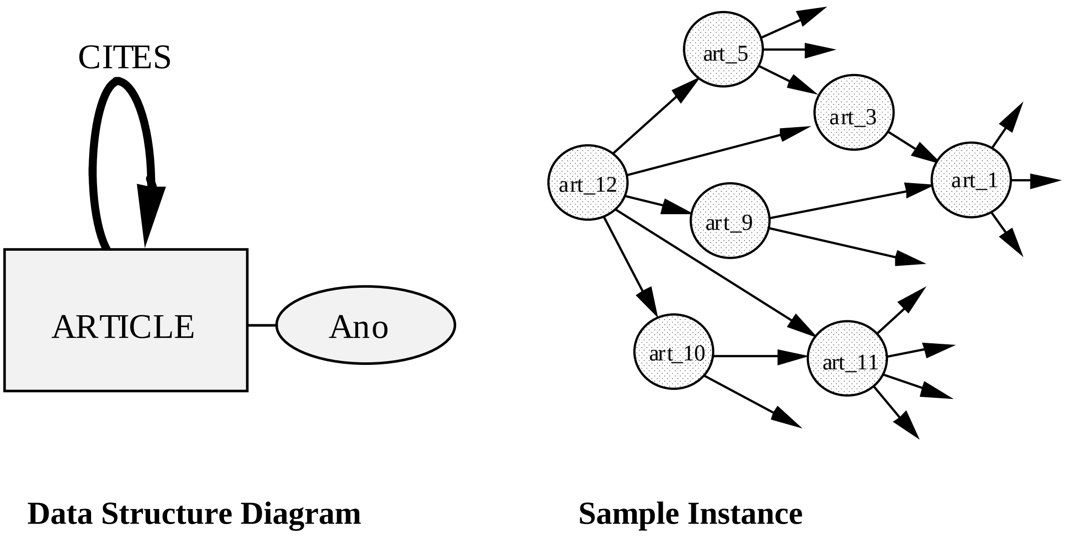


Fig. 2.3. Modeling a transitive non-hierarchic relationship

SUBTERM	
PREDECESSOR	**SUCCESSOR**
Data management	Data restructuring
Data management	Document retrieval
Data management	Query processing
Document retrieval	Query formulation
Document retrieval	Text retrieval
Document retrieval	Index construction

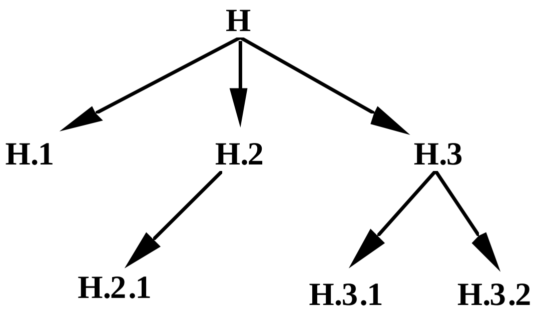
CITES	
PREDECESSOR	**SUCCESSOR**
art_12	art_5
art_12	art_3
art_12	art_9
art_12	art_11
art_12	art_10
art_5	art_3

Fig. 2.4. Representing transitive relationships as binary relations (partial)

Figure 2.4 shows how the transitive relationships are represented as binary relations. The columns are labeled as PREDECESSOR and SUCCESSOR. In the case of the hierarchic term relationship SUBTERM, predecessors give the hierarchically higher terms and successors the hierarchically lower terms. In the case of the citation relationship CITES, predecessors give the citing articles and successors the cited articles (NB: predecessor is later in time). The transitively hierarchically lower terms of, e.g., ‘Data management’ are denoted by successors(‘Data management’, [SUBTERM]) = {Data restructuring, Document retrieval, Query processing, Query formulation, Text retrieval, Index construction}. The first argument specifies the starting object and the second the binary relation as the context of transitive computation. Similarly, the immediate citing articles of, e.g., article art_1 are denoted by im_predecessors(art_1, [CITES]) = {art_3, art_9, ... }. The immediate cited articles (i.e., references) of article art_5 are denoted by im_successors(art_5, [CITES]) = {art_3, ...}. These notations correspond to the operations our query language for transitive processing (Niemi & Järvelin, 1992; Järvelin & Niemi, 1993) which will be used below.

The citation relationship CITES contains also self-citations, i.e., one of the authors of the citing document belongs to the authors of the cited document. In some analyses this would distort the statistics and therefore we sometimes use a subset CITES2 of the citation relationship CITES from which self-citations have been excluded.

Figure 2.5 shows a transitive hierarchic relationship, in this case the Computer Science Classification. All subclasses of, e.g., the class H.3 are denoted by successors(H.3, [SUBCLASS]) = {H.3.1, H.3.2}.



SUBCLASS	
PREDECESSOR	SUCCESSOR
H	H.1
H	H.2
H	H.3
H.2	H.2.1
H.3	H.3.1
H.3	H.3.2

Fig. 2.5. Representing transitive relationships of the CR Classification (partial)

In summary, an NF^2 relation-like complex object representation is not suitable for representing structures based on transitive relationships. Terminological relationships and citation (or other link) networks are not aggregation hierarchies in the data modeling sense (Smith & Smith, 1977). The networks consist of instances of objects (nodes) of a single type. In processing transitive relationships, the management of indirect node relationships is of prime importance, not the structure of nodes. Complex objects are needed when several separate objects with their own identity (e.g., relations) are put together to represent a complex real world entity, such as a document. In processing complex objects, the management of structural relationships is of prime importance. Because there is no static structure among subdocuments (component objects) in which all users would always want their result documents, a mechanism for restructuring the hierarchical relationships among subdocuments into new result documents is needed (Niemi & Järvelin, 1995).

In informetrics, complex objects (like bibliographic references), hierarchic transitive relationships (thesauri), as well as non-hierarchic transitive relationships (citation relationships) are all needed and often in combinations. In the sample database there are data for six object types which are typical foci of informetric analysis: authors, articles, journals, departments and their parent organizations as well as countries. We shall demonstrate that it is very easy for the user to obtain various informetric analyses of all these object types and, further, very easy to swap between the object types in the analyses. The database also contains several attributes typically used to select and/or break down the statistical data for the analysis of, e.g., possible trends: keywords, classification codes, publication years. In principle, nothing prevents using object types (e.g., journals) for data breakdowns and the breakdown attributes (e.g., years, classes) as the objects to analyze. The data represented by complex objects and transitive relationships are integrated through queries explained in the next section.

3. THE QUERY INTERFACE

So far, the query languages proposed for novel database paradigms have been too cumbersome to use from the viewpoint of end-users: users are required to derive the result data from the existing data by, often recursive, logical rules or constructors. Large nested expressions are usual in queries which combine data aggregation, transitive computation and data restructuring.

The idea behind the FUN interface is that all required data manipulation operations are deduced automatically on the basis a high-level declarative query specification. The user only has to express seven simple constructs in query formulation, when full aggregation, restructuring, transitive processing, sorting and retrieval capabilities are needed. The FUN interface has been described in earlier publications (Niemi & Järvelin, 1995; Järvelin & Niemi, 1995; Järvelin & Niemi, 1997). A query in the FUN interface is structured according to the following constructs:

- **form** <the form>
- **relations** <source relations>
- **conditions** <Boolean expression>
- **aggregation** <aggregation declaration>
- **subquery** <expression declaration>
- **sorting** <sort attributes>
- **printing**<output relations>

where

- the **form** construct is the linear schema representation of the result NF^2 relation,
- the **relations** construct is a list of names of existing (source) first normal form (1NF) or NF^2 relation(s) providing the source data for the query,
- the **conditions** construct is a Boolean expression which gives the filtering conditions of atomic-valued and relation-valued attributes,
- the **aggregation** construct is a list giving the aggregation way of the aggregated attributes,
- the **subquery** construct describes any transitive and other processing needed in the construction of each relation-valued result attribute

- the **sorting** construct is a list of atomic-valued attribute names used for sorting the result relation-valued attributes,
- the **printing** construct is a list of names of relation-valued attributes in the output.

The user gives these seven components in a straightforward way as exemplified below. Nothing else is required from the user. The query processing system deduces the retrieval, restructuring, aggregation and deductive operations needed for producing the result NF^2 relation from the source NF^2 relation(s). It also executes the expressions given in the **subquery**-component and applies the results according to the **condition** and/or the **form** constructs in the construction of the result. In the interface, the user specifies the schema level of the result NF^2 relation declaratively and the query processing system constructs its instance.

The FUN interface is structured in the conventional style, resembling the SQL. However, there are several differences with respect to the proposed SQL extensions (see, e.g., Pistor & Andersen, 1986; Roth & *al.*, 1987; Südkamp & Linnemann, 1990) for processing NF^2 relations: (i) our interface does not contain any explicit restructuring expressions — all restructuring is specified implicitly in the form; (ii) multi-attribute multi-way multi-level aggregation may be specified declaratively in a single query without nested expressions, and (iii) transitive processing is integrated conveniently through available high-level operations in the **subquery**-component. Therefore, queries in the FUN interface remain compact also when complex processing is required. We take the convention of presenting only those constructs of a query which specify processing. Therefore, e.g., the **conditions** construct is not presented when the condition is the plain 'true'. In fact, when not specified, the last five components may be omitted.

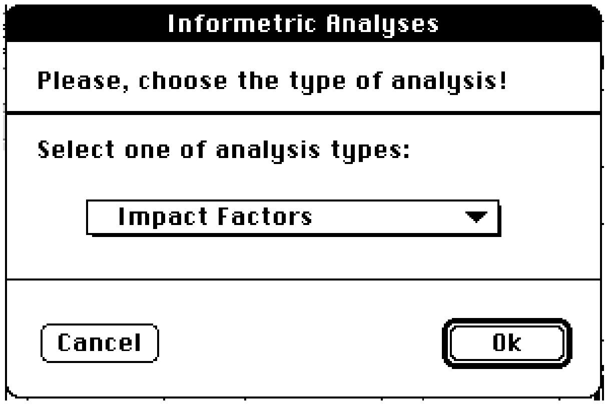
If query results are stored for later use, our interface assumes that the attribute names remain unique. Therefore renaming of attributes is applied, when necessary, using the **renaming** construct:

renaming {(OldName1, NewName1), ..., (OldNamek, NewNamek)}

which replaces the old attribute names by the new ones.

The query processing strategy and implementation issues are described by Niemi and Järvelin (1996; Järvelin & Niemi, 1997). The FUN interface has been implemented in LPA Prolog and runs on PCs and Macintoshes, as well as in Quintus Prolog for Unix machines. The sample query results in the following section are real output from the system using a small sample database.

In this paper we shall also present a user interface for informetric computation, which is based on online dialogs. Using this interface, the user need not directly use the query language presented above. This is important, because the high-level query language may still be too demanding for non-technical users and it is very easy to model repeating queries into simple online dialogs which fill in the variables for a query. Figure 3.1 presents the main menu dialog from which the user may choose various kinds of informetric analyses. Sample queries for informetrics in Section 4 will be presented both as query language expressions as well as online dialogs.



Informetric Analyses

Please, choose the type of analysis!

Select one of analysis types:

Impact Factors

Fig. 3.1. The main menu dialog

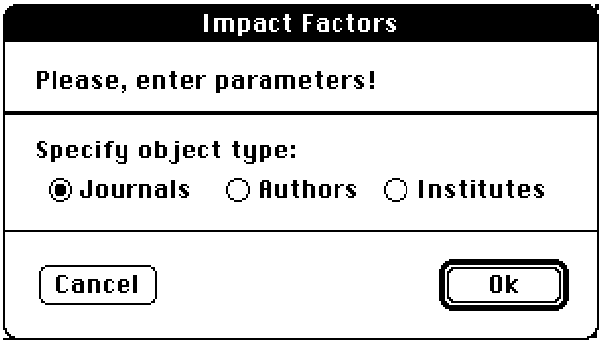
4. INFORMETRIC QUERIES

4.1. Generalized impact factors

Journal impact factors are among the most important and popular citation analytic measures (Egghe & Rousseau, 1990; Moed & van Leuwen, 1996). They are used, e.g., in the assessment of the expected scientific merit of scholars or research groups. The Journal Citation Report by ISI is a standard source for journal impact factors. Hjortgaard Christensen and Ingwersen (1996) demonstrate how various citation analyses of journals may be performed online, by using the Dialog retrieval system, for one or more volumes of a specific journal. The following remarks can be made concerning the state-of-the-art methodology presented by Wormell (1998):

- The user needs to process each journal separately.
- The user needs to specify each year of citation and publication separately.
- The resulting data require statistical post processing before the number of citations to each volume of each particular journal can be derived.

In this section we demonstrate how journal citation analyses, in particular journal impact factors, can be performed conveniently through the FUN interface. We shall also demonstrate how journal citation analyses are easily generalized to citation analyses of other object types, e.g., authors, institutions, countries, or classes of a classification. This is important since only authors, journals, and cited publication years at present can be analyzed directly for citation impact in the ISI citation databases. Figure 4.1 shows the dialog for impact factor analysis. By selecting the proper radio button, impact factors for journals, authors and institutes may be computed. Impact factors for classes have a separate dialog.



Please, enter parameters!

Specify object type:
● Journals ○ Authors ○ Institutes

Fig. 4.1. The main menu for impact factor analysis

Sample Expression 1

```
CITWINDOW =
          form          CITWINDOW(ano)
          relations     ARTICLES
          conditions    year = between([1988, 1995])

form          JOURNAL(journal,
                        PUBLYEAR(year, citation_sum, art_cnt,
                                  ARTS(ano, citation_cnt,
                                            CITATION(citing_art))))
relations     ARTICLES
conditions    year     1980 and year     1990
aggregation   citation_sum = sum(citation_cnt);
              citation_cnt = cnt(citing_art)
              art_cnt = cnt(ano);
subquery      CITATION(citing_art) =
                        set_intersection(
                                  im_predecessors(ano, [CITES]),
                                  CITWINDOW)
sorting       journal, year
printing      JOURNAL, PUBLYEAR
```

Fig. 4.2 (a) Sample Expression 1 for journal impact factor calculation

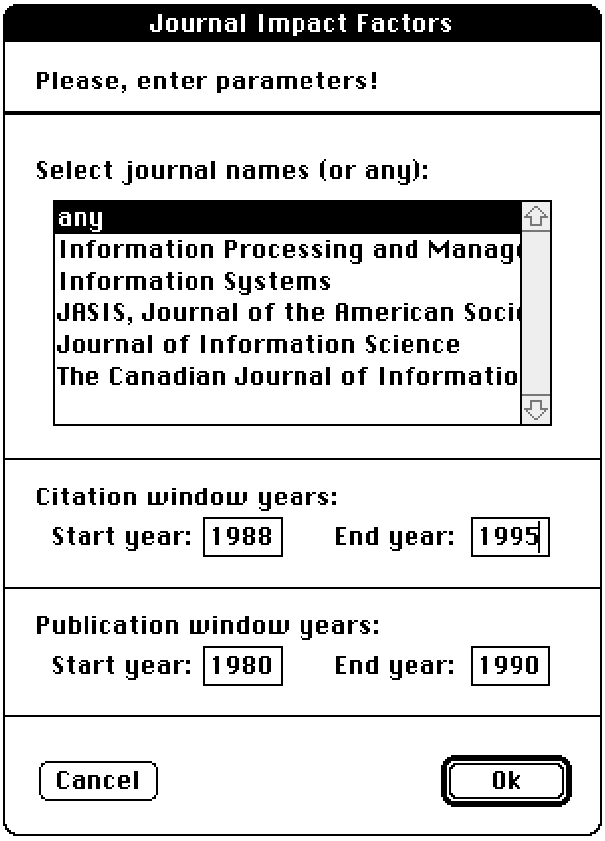
Sample Expression 1 (see Figures 4.2 a-b) has two queries to avoid nested expressions in the **subquery** construct. The first query limits the citation window (years of publication of citing articles) to desired years. In our sample case we use a relatively broad window (1988-1995), because the sample database is small. However, any window length can be used. This may often be a relevant way to generalize impact factors (Hjortgaard Christensen *& al.,* 1997). Also, any further conditions may be applied, e.g., the citing articles may be limited by journals, countries and/or disciplines. The **form** -construct determines that the result is a flat relation 'CITWINDOW' consisting only of article numbers published within the time range [1988, 1995]. Because only the three first components of the expression for 'CITWINDOW' specify any processing, the remaining components have been omitted.

The **form** construct of the main query specifies a data structure consisting of four levels of hierarchy. The top relation 'JOURNAL' gives journal names. For each journal, the relation-valued attribute 'PUBLYEAR' gives each year when the journal has published cited articles, together with citation statistics: the sum of received citations and the number of citable articles. Within each year, the relation-valued attribute 'ARTS' gives the citable articles of that year and the number of citations for each article. For each article number, the relation-valued attribute 'CITATION' identifies the citing articles. This relation-valued attribute is constructed by the **subquery** construct (see below). In this form the atomic-valued attributes 'journal', 'year' and 'ano' are *source relation attributes* and the rest *derived attributes.* Among the latter, 'citation_sum', 'art_cnt' and 'citation_cnt' are *aggregated attributes* and 'citing_art' a *deductive attribute* derived through a subquery.

The **conditions** construct of the main query specifies the publication window of the cited articles as the years within the range [1980, 1990]. The user may express any other conditions concerning the source relation attributes and/or derived attributes, e.g., conditions on cited article topics, cited journal names, etc. The **aggregation** construct states that the values of the aggregated attribute 'citation_cnt' is a count on the values of the attribute 'citing_art', the aggregated attribute 'citation_sum' is a sum of the values of the attribute 'citation_cnt', and the aggregated attribute 'art_cnt' is a count on the values of the attribute 'ano'. Thus multiple attributes are aggregated at two levels at once.

The **subquery** constructs the relation-valued attribute 'CITATION'. The left-hand side of the expression is the form of the relation-valued attribute and the right-hand side expresses its derivation. The expression im_predecessors(ano, [CITES]) finds all articles citing the individual articles (each identified by the 'ano' -value) for which the relation-valued attribute 'CITATION' is being constructed. The result is a set of citing article numbers. The NF^2 relation name 'CITWINDOW' denotes the whole 'CITWINDOW' relation, the (citing) article numbers of which are returned as a set (see Järvelin & Niemi, 1993). The two sets of article numbers are finally intersected by the operation set_intersection. This yields a set of article numbers for articles, which cite the article under consideration and are published within the required citation window 1988-95. One should note that in formal scientific communication an *article* is only *cited once* on a reference list. However, the *journals* in question can be cited *several times* by the same article.

The **printing** construct of the main query specifies that only the two top relation-valued attributes 'JOURNAL' and 'PUBLYEAR' are reported as the result. The other relation-valued attributes are, in fact, only needed for computing the aggregated attributes and can therefore be omitted from the result. The **sorting** construct specifies that the relation-valued attribute 'JOURNAL' is sorted on journal names and the relation-valued attribute 'PUBLYEAR' on years.



Journal Impact Factors

Please, enter parameters!

Select journal names (or any):

any
Information Processing and Manag
Information Systems
JASIS, Journal of the American Soci
Journal of Information Science
The Canadian Journal of Informatio

Citation window years:
Start year: 1988 End year: 1995

Publication window years:
Start year: 1980 End year: 1990

Fig. 4.2 (b) The online dialog for standard journal impact factors

Figure 4.2b presents the standard dialog for journal impact factors. It allows impact factors calculation for any set of named journals as chosen by the user, or all journals in the database. The citation and publication window years can also be given. It is easy to modify these dialogs to accommodate other frequent parameters, e.g., publishers, countries or scientific domains, when needed. The set of journal names in the menu is constructed by a query in the FUN query language.

```
{(Information Processing and Manageme,
        {(1990, 1, 1)}),
 (Information Systems,
        {(1981, 1, 1),
         (1986, 1, 1),
         (1987, 2, 1)}),
 (JASIS, Journal of the American Soci,
        {(1980, 4, 1)}),
 (Journal of Information Science,
        {(1987, 1, 1)}),
 (The Canadian Journal of Information,
        {(1984, 1, 1)})}
```

Fig. 4.2 (c) Sample Expression 1 result: Impact factors for journals

The query result (Figure 4.2c) gives the 7-year synchronic impact data for five journals and various individual years within the range of 1980-90. For example, during the period 1988-95 JASIS has received four citations for one article published in 1980 (remember that the database is small).

Sample Expression 1 has several salient features:

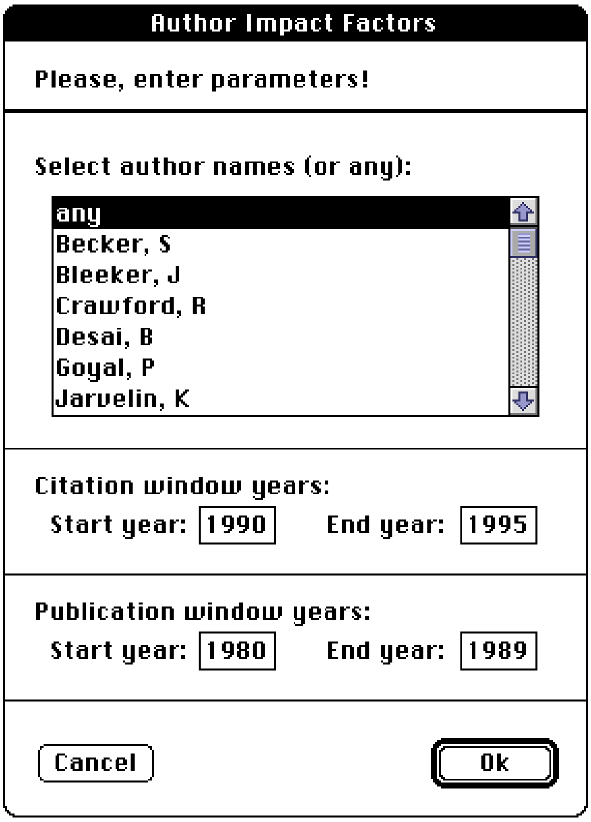
- The user need not process each journal separately. Instead, she gets data for all relevant journals automatically. Note that the **condition** construct could contain any conditions directly on journal names, publishers, countries of publication, and/or scientific domains combined with either the cited or the citing articles or both.
- The user need not specify each year of citation separately. Instead, she gets data for all relevant years automatically.
- The resulting data give the sum of citations as well as the number of citable articles directly for impact factor calculation. If summations over the publication years for each journal are needed, these are easily derived by defining two new attributes, e.g., sum_of_citations and sum_of_articles.
- Multi-level multi-attribute aggregation is performed in a single query.

When the properties of citing and/or cited documents are used in the query, these documents must be included in the database as fully represented documents. In practice, all databases contain documents, which either give or receive citations across the database boundaries and thus the citing or cited documents are external to the database. However, this limitation affects all approaches to citation analysis.

Through the FUN interface, it is very simple to obtain data for various generalizations of impact factors. For example, for *author impact factors,* it is sufficient just to change the **form**, **sorting** and **printing** constructs as follows (changes in *italics*):

```
form        AUTHOR(author, citation_sum, art_cnt,
                   ARTS(ano, citation_cnt,
                          CITATION(citing_art))))
sorting     citation_sum
printing    AUTHOR
```

The user need not do anything else and therefore it is very easy for the user to analyze the data in various ways (the computer may be busy, though). In this case we left out the data breakdown by years simply by dropping the relation-valued attribute ‘IMPACTYEAR’ and the attribute ‘year’, and by moving the aggregated attributes by one level up.



Author Impact Factors

Please, enter parameters!

Select author names (or any):

any
Becker, S
Bleeker, J
Crawford, R
Desai, B
Goyal, P
Jarvelin, K

Citation window years:
Start year: 1990 End year: 1995

Publication window years:
Start year: 1980 End year: 1989

Fig. 4.3 (a) The online dialog for standard author impact factors

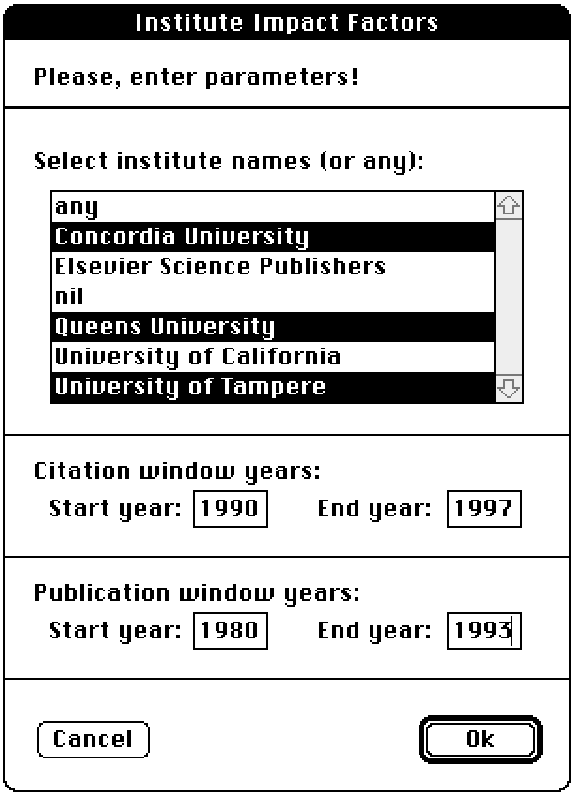
By selecting “authors” in the dialog of Figure 4.1, the dialog of Figure 4.3a for standard impact factors query for authors is presented. The user may specify the author set and the citation and publication window years. Here all authors are selected for the citation window 1990-95 and the publication window 1980-98. From the same source data as above, the result is as given in Figure 4.3b. For example, Sadri has received two citations during the period 1990-95 for the article he has in the database (published 1980-89).

```
{(Becker, S, 1, 1),
 (Bleeker, J, 1, 1),
 (Crawford, R, 5, 2),
 (Desai, B, 2, 1),
 (Goyal, P, 2, 1),
 (Kircz, J, 1, 1),
 (Macleod, I, 1, 1),
 (Ogilvie, J, 1, 1),
 (Sadri, F, 2, 1),
 (Scheck, H, 1, 1),
 (Scholl, M, 1, 1)}
```

Fig. 4.3 (b) Impact factors data for authors

For institutional impact factors, it is similarly sufficient just to change the **form**, **sorting** and **printing** constructs as follows (changes in *italics*):

form *INSTIT(organization,* citation_sum, art_cnt,
ARTS(ano, citation_cnt,
CITATION(citing_art))))
sorting *organization*
printing *INSTIT*



Institute Impact Factors

Please, enter parameters!

Select institute names (or any):

any
Concordia University
Elsevier Science Publishers
nil
Queens University
University of California
University of Tampere

Citation window years:
Start year: 1990 End year: 1997

Publication window years:
Start year: 1980 End year: 1993

Fig. 4.4 (a) The online dialog for standard institutional impact factors

The online dialog for standard institutional impact factors, based on this query modification, is presented in Figure 4.4a. Here three universities are selected and citations during 1990-97 to their publications in 1980-93 are calculated. From the same source data, the result is as given in Figure 4.4b. For example, Queens University has received eight citations during the period 1990-97 for the five articles by the university in the database in 1980-93.

```
{(Concordia University, 2, 1),
 (University of Tampere, 4, 3),
 (Queens University, 8, 5)}
```

Fig. 4.4 (b) Impact factors data for organizations

One may notice that, in contrast to the proposed NF^2 relational model, institutional or national impact factors cannot be obtained in contemporary CD-ROM or online citation indexes directly. They are only available through cumbersome selection of individual cited documents authored by the institution or country.

In a very similar way, the impact factors can be computed for disciplines (if journals have discipline codes), topical classes, keywords, etc. Therefore it may be concluded that it is very easy to get various impact factors by simply manipulating the **form** construct. The required data breakdowns are obtained by placing source attributes in suitable positions within relation-valued attributes of the form.

As a final example on impact factors, the impact factors of classes used to index information retrieval literature can be computed by the Sample Expression 1b (modifications in *italics*). The expression and its result are given in Figures 4.5a-c. In this expression, IR literature is identified in the **condition** construct through articles having one of its 'key' attributes belonging to IR_KEYS. The latter are defined by the first subquery IR_KEY(irkey) = successors('document retrieval', [subterm]) as keys which are subterms of 'document retrieval'. Note that this subquery traverses the term hierarchy transitively toward subterms. This leads to query expansion (e.g., Kekäläinen & Järvelin, 1998). We have demonstrated in (Järvelin & Niemi, 1993) how multiple such hierarchies may be traversed upwards and downwards in a single transitive expression. The condition could, of course, be given directly on subclasses of the CS Class 'H.3' (information storage and retrieval) by the **condition** class = one_of(IR_CLASS(class)) and the **subquery** IR_CLASS(class) = successors('H.3', [subclass]). The CITWINDOW relation is the same as in Figure 4.2a (but for the correct years).

```
Sample Expression 1b
form          IR_CLASS(class, citation_sum, art_cnt,
                      ARTS(ano, citation_cnt,
                           CITATION(citing_art))))
relations     ARTICLES
conditions    key = one_of(IR_KEY(irkey)) and year = between([1980, 1989])
aggregation   citation_sum = sum(citation_cnt);
              citation_cnt = cnt(citing_art)
              art_cnt = cnt(ano);
subquery      IR_KEY(irkey) = successors('document retrieval', [subterm]);
              CITATION(citing_art) =
                    set_intersection(
                          im_predecessors(ano, [CITES]),
                          CITWINDOW)
sorting       class
printing      IR_CLASS
```

Fig. 4.5 (a) The expression for impact factors of classes

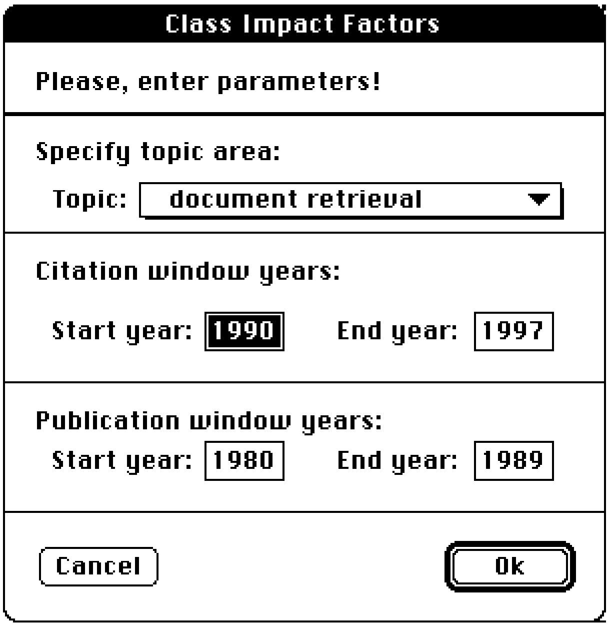
The query result in Figure 4.5c shows those Computer Science classes which occur in the articles in the document retrieval area. The statistics give, first, the number of citations to each class (to an article classified in each class) during 1990-97 and, second, the number of articles for each class during 1980-89.

The data for the immediacy index, another popular informetric measure, and its generalizations may be obtained in a very similar way.

4.2. Author co-citation analysis

Author co-citation analysis (ACA) is an established area of informetrics (e.g., White, 1990). McCain (1990) gives a technical overview of the procedures required in ACA. In a traditional ACA data collection, co-citation counts are collected for each pre-selected pair of authors through a range of separate queries. These co-citation counts are then arranged into a raw co-citation matrix for further analysis, for instance, in order to generate maps of a scientific domain by means of multi-dimensional scaling (MDS). There are further complications in data collection if co-authors in the second author position or beyond are to be taken into account.

In this section we shall demonstrate, how the raw data for ACA and its generalizations can be computed declaratively through the FUN interface. However, we shall first consider ACA (see Figure 4.6a) and then the generalizations (institution and class cocitation analysis).



Class Impact Factors

Please, enter parameters!

Specify topic area:

Topic: document retrieval

Citation window years:

Start year: 1990 End year: 1997

Publication window years:

Start year: 1980 End year: 1989

Fig. 4.5 (b) The online dialog for standard impact factors for selected ACM CS classes

```
{(H.2.1, 1, 1),
 (H.2.2, 1, 1),
 (H.2.4, 2, 2),
 (H.3.2, 1, 1),
 (H.3.3, 2, 2),
 (H.4.1, 1, 1)}
```

Fig. 4.5 (c) Impact factors for selected ACM CS classes

In Figure 4.6a we first construct an auxiliary NF^2 relation ‘ARTICLES2’ giving data on articles and their authors. This is to avoid nested subqueries. The NF^2 relation ‘ARTICLES2’ is renamed and stored with the form ARTICLES2(cc_ano, cc_author). This relation is used to identify the co-cited articles and their authors, therefore the attribute names ‘cc_ano’ and ‘cc_author’. Note that any further attributes may be included, as needed, into the result relation. The **conditions** construct in the sample expression selects articles in the subclasses of the ACM CS class ‘H’ published in 1990-97. Any further selection criteria, e.g., publishing journals, or their combinations may be used.

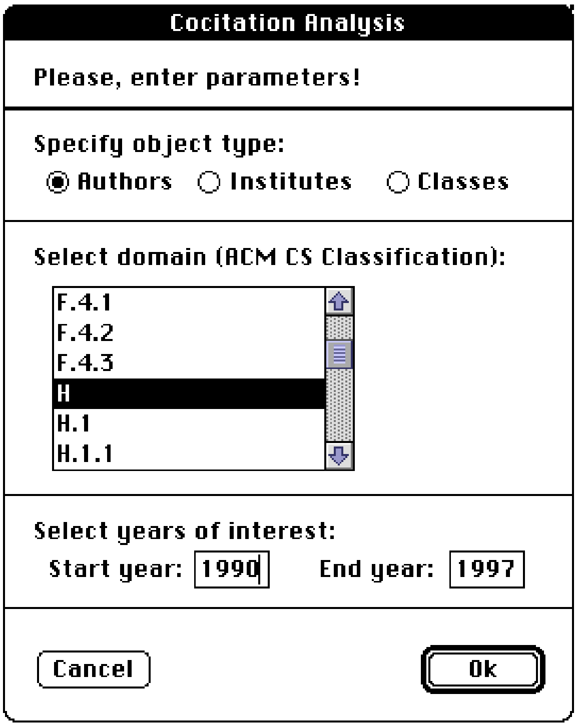
Sample Expression 2

```
ARTICLES2 =
          renaming      {(ano, cc_ano), (author, cc_author)}
          form          ARTICLES2(ano, author)
          relations     ARTICLES
          conditions    class = in(DOMCLASSES(domclass)) and
                        year = between([1990, 1997])
          subquery      DOMCLASSES(domclass) =
                                  union_of_successors([H], [subclass]))

form          AUTHOR_COCITATIONS(author, cc_author, cocicosum, sum_citing1,
                        sum_citing2,
                        COCIT_ARTS(ano, cc_ano, cocico, no_citing1, no_citing2,
                                  'CIT1'(c_ano1),
                                  'CIT2'(c_ano2),
                                  JOINTCITS(jc_ano)))
relations     ARTICLES, ARTICLES2
conditions    class = in(DOMCLASSES(domclass)) and
              year = between([1990, 1997]) and
              cocicosum > 2 and author ≠ cc_author and ano ≠ cc_ano
aggregation   cocicosum = sum(cocico);
              cocico = cnt(jc_ano);
              sum_citing1 = sum(no_citing1);
              sum_citing2 = sum(no_citing2);
              no _citing1 = sum(c_ano1);
              no _citing2 = sum(c_ano2)
subquery      CIT1(c_ano1) = im_predecessors(ano, [CITES2])),
              CIT2(c_ano2) = im_predecessors(win_ano1, [CITES2])
              JOINTCITS(jc_ano) =
                        set_intersection(
                                  im_predecessors(ano, [CITES2]),
                                  im_predecessors(cc_ano, [CITES2]))
              DOMCLASSES(domclass) =
                        union_of_successors([H], [subclass]))
sorting       author
printing      AUTHOR_COCITATIONS
```

Fig. 4.6 (a) Sample Expression 2 for author co-citations

The query for author co-citation data uses the original NF^2 relation ‘ARTICLES’ and the auxiliary relation ‘ARTICLES2’. To avoid the effects of self-citations in combination of multiple authors, this query utilizes the citation network CITES2 from all self-citations have been removed. The query considers first each article in ‘ARTICLES’ and finds for each article in ‘ARTICLES2’ whether these two have been co-cited. The articles in ‘ARTICLES’ could be selected by any criteria — as in the case of ‘ARTICLES2’. The co-citing articles (i.e., articles with article numbers jc_ano citing both current ‘ano’ and ‘cc_ano’) are computed into the relation-valued attribute JOINTCITS(jc_ano) in the **subquery** construct. The subquery identifies citing articles for both articles under consideration and takes the intersection of the two sets. In order to normalize the cocitation data, the citations for each article individually are computed into the relation-valued attributes CIT1(c_ano1) and CIT2(c_ano2). The non-transitive look-ups, e.g., im_predecessors(ano, [CITES2]), are very fast to execute because only a very small subset of the citation network is considered.

The **form** construct uses JOINTCITS(jc_ano) for aggregating the co-citation count ‘cocico’ for each article pair (‘ano’ and ‘cc_ano’) in the relation-valued attribute ‘COCIT_ARTS’. The latter gives for each article (ano) all co-cited articles with their article numbers (cc_ano), authors and co-citation counts, as well as the citation counts of the articles individually. In the top relation ‘AUTHOR_COCITATIONS’ the aggregated attribute ‘cocicosum’ gives the author co-citation strength as a simple sum of the authors’ co-citations. The aggregated attributes sum_citing1 and sum_citing2 give the sums of citations for each author individually. The **condition** construct prunes out all author pairs which have little co-citations — cocicosum > 0. The conditions author ≠ cc_author and ano ≠ cc_ano prevent reporting a person’s co-citations with him/herself and counting co-citations for one article with itself. The domain and the publication years are restricted as for the relation ARTICLES2. Any further conditions may be used to restrict the set of articles for which co-citations are considered.


Cocitation Analysis

Please, enter parameters!

Specify object type:

● Authors ○ Institutes ○ Classes

Select domain (ACM CS Classification):

F.4.1
F.4.2
F.4.3
H
H.1
H.1.1

Select years of interest:

Start year: 1990 End year: 1997

Fig. 4.6 (b) The online dialog for standard author cocitation analysis

The **sorting** and **printing** constructs organize the result as a flat relation sorted by authors and giving for each the co-cited authors and co-citation counts.

Figure 4.6b presents the online dialog for standard author cocitation analysis with selections matching Sample Expression 2. In the dialog the user may choose author, institute or class cocitation analysis. The menu of ACM CS classes is produced automatically from the database and any class selections automatically include any subclasses into the analysis.

```
{...,
 (Lynch, C, Macleod, I,   2, 4, 2),
 (Lynch, C, Desai, B,     1, 2, 2),
 (Lynch, C, Sadri, F,     1, 2, 2),
 (Lynch, C, Goyal, P,     1, 2, 2),
 (Lynch, C, Scheck, H,    1, 2, 1),
 (Lynch, C, Scholl, M,    1, 2, 1),
 (Lynch, C, Kircz, J,     1, 2, 1),
 (Lynch, C, Bleeker, J,   1, 2, 1),
 (Macleod, I, Desai, B,   1, 1, 2),
 (Macleod, I, Sadri, F,   1, 1, 2),
 (Macleod, I, Goyal, P,   1, 1, 2),
 (Macleod, I, Scheck, H,  1, 1, 1),
 (Macleod, I, Scholl, M,  1, 1, 1),
 (Macleod, I, Kircz, J,   1, 1, 1),
 (Macleod, I, Bleeker, J, 1, 1, 1),
 (Macleod, I, Lynch, C,   2, 2, 4),
...}
```

Fig. 4.6 (c) Sample Expression 2 result for author co-citation analysis

Figure 4.6c presents the resulting data, which may be submitted to further ACA processing, e.g., for producing author clusters and maps. It is straightforward to use such data as an input file to MDS for further analysis, as recently done on information science by White & McCain (1998). The sample data indicate that, e.g., Macleod and Lynch have been co-cited twice for two and four individual citations.

Again, salient features of our expressions are, among others, that the user need not form, in advance, retrieved sets for each cited author and then produce the co-citation data for each pair of authors separately. Instead, the co-cited authors are found within the data. Moreover, all authors of cited papers are treated equally. (The traditional way of ACA, focusing on first authors, could be made by modeling article authors as an atomic-valued first author and a relation-valued ‘COAUTHOR’ set.).

The user can easily navigate in the data structures and produce the data breakdowns and aggregations relevant in her current situation. For example, she obtains institutional co-citation data simply by replacing the authors by their institutions in the **form** constructs as follows (modifications in *italics*).

ARTICLES2 =

    renaming {(ano, cc_ano), (*organization, cc_organization*)}

    form ARTICLES2(ano, *organization*)

    ...

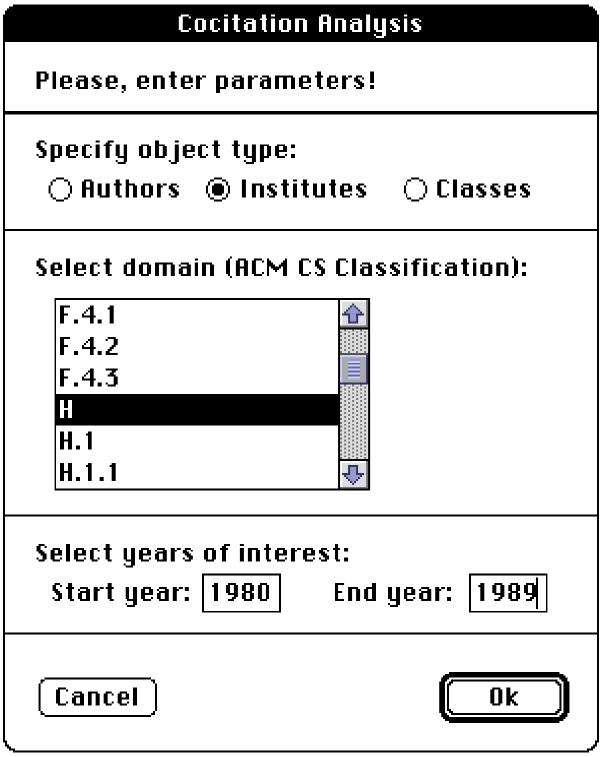
form *ORG_COCITATIONS(organization, cc_organization,* cocicosum, …

    …

conditions cocicosum > 0 **and** *organization ≠ cc_organization* **and** ano ≠ cc_ano

...

If the **printing** and **sorting** constructs are used, they need updating for proper attribute names. These modifications are effected by the selection of the radio button for institutes in the on-line dialog for standard cocitation analysis (see Figure 4.7a). The result of this query is given in Figure 4.7b, which shows that Queens University and Concordia University have been co-cited three times.



Cocitation Analysis

Please, enter parameters!

Specify object type:

○ Authors ● Institutes ○ Classes

Select domain (ACM CS Classification):

F.4.1
F.4.2
F.4.3
H
H.1
H.1.1

Select years of interest:

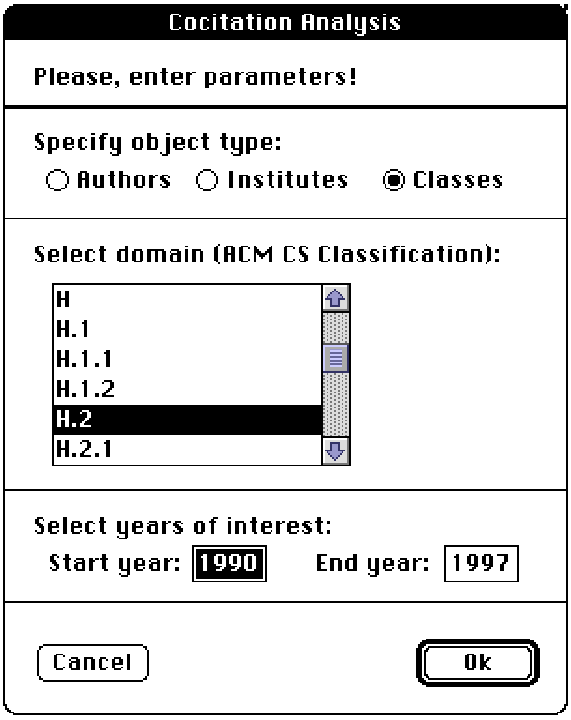
Start year: 1980 End year: 1989

Fig. 4.7 (a) The online dialog for standard institutional cocitation analysis

```
{(Concordia University, Queens University,          3, 4, 8),
 (Concordia University, Elsevier Science Publishers, 1, 2, 1),
 (Elsevier Science Publishers, Concordia University, 1, 1, 2),
 (Elsevier Science Publishers, Queens University,    2, 2, 8),
 (Queens University, Concordia University,          3, 8, 4),
 (Queens University, Elsevier Science Publishers,    2, 8, 2)}
```

Fig. 4.7 (b) Sample result for institutional co-citation data (partial)

Figures 4.8a and 4.8b show the dialog and the result for class co-citation analysis. It has been computed for some database management and IR classes of the ACM CS Classification (H.2 and H.3 with subclasses) and only reports classes co-cited at least two times. The structure of the result is as in the preceding cocitation cases. This kind of data may be used, among others, for statistical clustering of classes (of keywords, etc.) and relevant measures of nearness or interdisciplinarity across scientific fields. In contemporary systems, class cocitation analysis would require downloading and combining data from citation indexes (e.g., ISI indexes) and traditional bibliographic databases.



Cocitation Analysis

Please, enter parameters!

Specify object type:
○ Authors ○ Institutes ● Classes

Select domain (ACM CS Classification):
H
H.1
H.1.1
H.1.2
H.2
H.2.1

Select years of interest:
Start year: 1990 End year: 1997

Fig. 4.8 (a) The online dialog for standard class cocitation analysis

In summary, our approach provides several benefits for co-citation analysis and its generalizations:

- the user need not have a predefined list of the objects of interest to start with (picking those which will score high in the statistics may require considerable experience in the area);
- the user need not collect the raw data by forming queries pairwise for the objects (articles, authors, organizations, countries, keywords) of interest;
- the objects of interest can automatically be selected through predicates
- the user can very easily change focus, the objects of interest.

```
{(H.2.3, H.3.3, 2, 4, 2),
 (H.2.4, H.3.3, 3, 5, 4),
 (H.2.4, H.3.2, 2, 3, 3),
 (H.3.2, H.2.4, 2, 3, 3),
 (H.3.2, H.3.3, 3, 5, 4),
 (H.3.3, H.2.3, 2, 2, 4),
 (H.3.3, H.2.4, 3, 4, 5),
 (H.3.3, H.3.2, 3, 4, 5)}
```

Fig. 4.8 (b) Sample result for class co-citation data

4.3. Recognized contribution

White (1990) mentions the possibility of replacing author points in an author co-citation map by three or four expressions appearing most frequently in the titles of articles citing each author. One may say that these expressions reflect, statistically, the issues and topics for which each author has produced a recognized contribution. In this section we demonstrate, how such information may be computed in the FUN interface.

We shall illustrate White's idea by using the keywords of citing articles as content indicators for author contributions. (This is because the sample database contains article titles as whole strings. When setting up the database, the titles could, of course, be formed as sets of words and phrases in a relation-valued attribute TITLEWORDS(word). That would allow the computation of exactly what White projected.).

Sample Expression 3 (Figure 4.9a-b) utilizes two pre-computed auxiliary relations, 'CITEDART' and 'ARTICLEKEYS'. The former gives, for each article, the articles which cite it, and the latter for each (citing) article within the required citation window its keyword set. Renaming of attributes is done to reflect the role of each attribute in the final query.

The main query expression joins the NF^2 relations 'ARTICLES', 'CITEDART' and 'ARTICLEKEYS' so that each article as a cited article (the condition ano = cited_ ano) is associated with the its citing articles and their keywords (the condition citing_ano = citing_ano2). The

form construct arranges the data, giving for each author the citing keywords and their frequency ‘key_count’ in any articles citing any of his/her articles. Thus each author is ‘indexed’ by his/her recognized contribution. The **conditions** construct contains the join conditions for the NF^2 relations and the selection condition for the authors of interest as well. A threshold may be given on the keyword frequencies by (key_count > *threshold*). Keywords with lower frequencies would thus not be reported.

Sample Expression 3

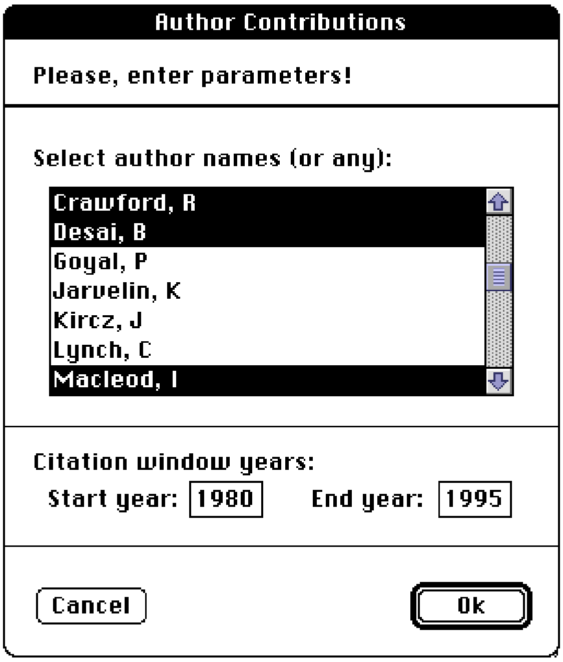
```
CITEDART =
        renaming     {(ano, cited_ ano), (citing_ano, citing_ano)}
        form         CITEDART(ano,
                             CITINGART(citing_ano))
        relations    ARTICLES
        subquery     CITINGART(citing_ano) =
                             set_intersection(
                                     im_predecessors(ano, [CITES]),
                                     WINDOW)

ARTICLEKEYS =
        renaming     {(ano, citing_ano2), (keyword, citing_keyword)}
        form         ARTICLEKEYS(ano, KEYS(keyword))
        relations    ARTICLES
        conditions   year = between([1980, 1995])

form         AUTHOR_CONTRIBUTION(author,
                     CONTRIB_KEY(citing_keyword, key_count,
                             CITINGARTS(citing_ano)))
relations    ARTICLES, CITEDART, ARTICLEKEYS
conditions   ano = cited_ ano and citing_ano = citing_ano2 and
             author = in({'Crawford, R', 'Desai, B', 'Macleod, I'})
aggregation  key_count = cnt(citing_ano)
sorting      author, key_count
printing     AUTHOR_ CONTRIBUTION, CONTRIB_KEY
```

Fig. 4.9 (a) Sample Expression 2 for recognized contribution

The query can be presented in a simple form by the online dialog for standard recognized contribution analysis, see Fig. 4.9b. Here the menu of authors is computed by a query from the database. It is easy to see that further conditions, e.g., for selection authors in given disciplines, may easily be added to the online dialog. The query result in Figure 4.9c informs that, e.g., Crawford is known for contributions in relational databases and document retrieval.

As above, it is straightforward to obtain similar figures for organizations or countries by simple modifications in the **form** construct. We have defined also online dialogs for standard recognized contribution analysis for institutions, journals and countries (bypassed here). In a very similar way one may compute the scientific export to other fields:

- by selecting top journals in some area and finding source articles in these journals;
- by finding other articles which cite the source articles;
- by checking the classification codes of citing articles (or of the journals, if available);
- and by aggregating the classification codes.



Author Contributions

Please, enter parameters!

Select author names (or any):

Crawford, R
Desai, B
Goyal, P
Jarvelin, K
Kircz, J
Lynch, C
Macleod, I

Citation window years:
Start year: 1980 End year: 1995

Fig. 4.9 (b) The online dialog for standard recognized contribution analysis

```
{(Crawford, R,
     {(hierarchical objects, 1),
      (SGML, 1),
      (structured documents, 1),
      (query languages, 1),
      (bibliographic database, 1),
      (lazy evaluation, 1),
      (nonmaterialized relation, 1),
      (SQL, 1),
      (data restructuring, 1),
      (NF2 database, 1),
      (nf2 relation, 1),
      (query formulation, 1),
      (nest operation, 1),
      (non-first normal form relation, 1),
      (text retrieval, 2),
      (document retrieval, 2),
      (universal relation, 2),
      (relational database, 3)}),
 (Desai, B,
     {(data restructuring, 1),
      ...}),
 (Macleod, I,
     {(data restructuring, 1),
      (document retrieval, 1),
      (NF2 database, 1),
      ...})}
```

Fig. 4.9 (c) Sample result for recognized contribution (partial)

4.4. International visibility and international impact

Hjortgaard Christensen & Ingwersen (1996; Ingwersen & Hjortgaard Christensen, 1997) suggest *international visibility* and *international impact* of journals as journal evaluation criteria. These concepts can be approached from the quantitative point of view by considering the geographical distribution of the origin of articles of each journal and by the geographical scatter of article users (through citations). The contemporary online methods, as reported by Hjortgaard Christensen and Ingwersen, require separate consideration of each journal. In this section we shall demonstrate how these geographical distributions can be obtained through the FUN interface.

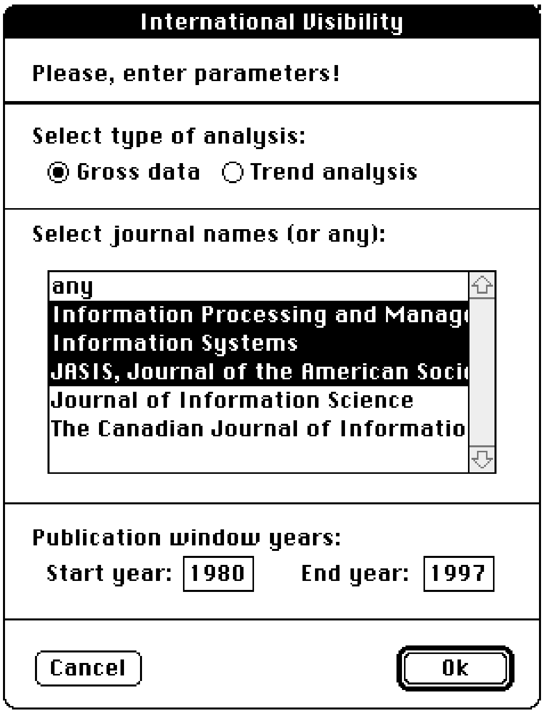
Sample Expression 4 (Figure 4.10a) computes the statistics on author home countries. Note that in case of multiple authors the home country of each author is included in the statistics; if there are multiple authors from one country in any single article, then this country is credited for each co-author each time they publish. Although the **conditions** construct only contains conditions on journal names and publication years, any conditions may be used to select countries, topical areas, etc. If the attribute author was dropped from the relation-valued attribute ARTS, then multiple co-authors from one country would credit the country statistics only by one.

Sample Expression 4	
form	JOURNAL(journal, art_total, AU_ORIGIN(country, art_cnt, ARTS(ano, author)))
relations	ARTICLES
conditions	journal = **in**('Information Processing …', 'Information Systems', 'JASIS, …') **and** year = **between**([1980, 1997])

aggregation	art_cnt = **cnt**(ano);
	art_total = **sum**(art_cnt)

sorting	journal, art_cnt
printing	JOURNAL, AU_ORIGIN

Fig. 4.10 (a) Sample Expression 4 on author geographical distribution



International Visibility

Please, enter parameters!

Select type of analysis:
● Gross data ○ Trend analysis

Select journal names (or any):

any
Information Processing and Manag
Information Systems
JASIS, Journal of the American Soci
Journal of Information Science
The Canadian Journal of Informatio

Publication window years:
Start year: 1980 End year: 1997

Fig. 4.10 (b) The online dialog for standard author geographical distributions

```
{(Information Processing and Manageme, 5,
     {(Canada, 1),
      (Finland, 4)}),
 (Information Systems, 8,
     {(Germany, 2),
      (Finland, 2),
      (Canada, 4)}),
 (JASIS, Journal of the American Soci, 5,
     {(USA, 1),
      (Finland, 2),
      (Canada, 2)})}
```

Fig. 4.10 (c) Sample result on author geographical distribution

The online dialog for standard author geographical distributions is given in Figure 4.10b. It corresponds to Sample Expression 4. The selection of the radio button 'Trend analysis' would modify the form construct by adding publication years a relation-valued attribute between the journals and the author origins. This would yield the author origin data by years. The result is given in Figure 4.10c. For example, Information Systems has altogether 8 co-authorships over the years with two from Finland and four from Canada.

In order to study possible trends in the distribution of author origins, one may break the data down by years by simply modifying the **form** (see in *italics*) :

```
form        JOURNAL(jname, art_total,
                    ANNUAL_DISTR(year,
                            AU_ORIGIN(country, art_cnt,
                                    ARTS(ano))))
printing    JOURNAL, ANNUAL_DISTR, AU_ORIGIN
```

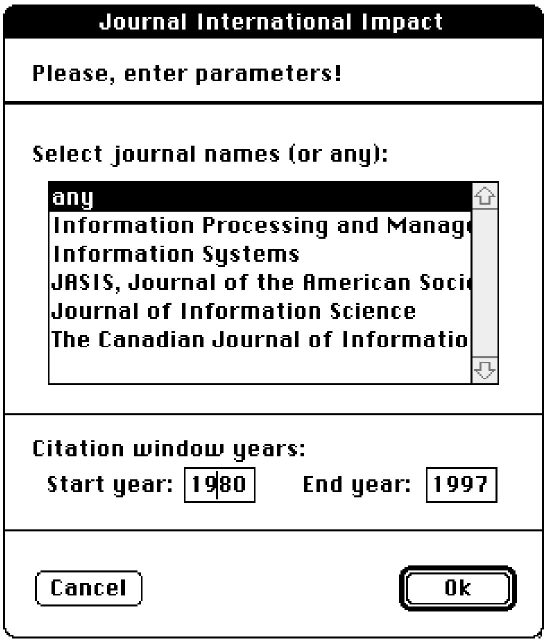
The geographical scatter of article users is computed by Sample Expression 5 (Figure 4.11a-b). To avoid nested expressions, we use two pre-computed auxiliary relations: ‘CITEDART’ (see Sample Expression 3) and ‘CITINGCOUNTRIES’. The latter gives for each article its country of origin (multiple countries, if there are multiple authors from different countries).

Sample Expression 5

```
CITINGCOUNTRIES =
        renaming    {(ano, c_ano), (country, c_country)}
        form        CITINGCOUNTRIES(c_ano, c_country)
        relations   ARTICLES
        conditions  year = between([1980, 1997])

form        JOURNAL(journal,
                    CITINGCOUNTRIES(c_country, citation_sum,
                            CITED_ARTS(ano, citation_cnt,
                                    CITING_ARTS(c_ano))))
relations   ARTICLES, CITINGCOUNTRIES, CITEDART
conditions  ano = cited_ano and citing_ano = c_ano
aggregation citation_sum = sum(citation_cnt);
            citation_cnt = cnt(c_ano)
sorting     journal, citation_sum
printing    JOURNAL, CITINGCOUNTRIES
```

Fig. 4.11 (a) Sample Expression 5 for the geographical scatter of citations to journals

The main query joins the three NF^2 relations ‘ARTICLES’, ‘CITEDART’ and ‘CITING-COUNTRIES’ on the attributes corresponding to article numbers (ano = cited_ano and citing_ano = c_ano). The **form** construct then rearranges the data to report for each journal its citing countries (‘c_country’), for each citing country the cited articles (‘ano’), and for each cited article the citing articles (‘c_ano’). Data **aggregation** goes up from the bottom relation CITING_ARTS. First the citing article numbers are counted into ‘citation_cnt’ in ‘CITED_ARTS’ and further, summed into ‘citation_sum’ in ‘CITINGCOUNTRIES’. Thus each citing country obtains the sum of citations originating from it. The two topmost relation-valued attributes are **printed**, **sorted** on ‘journal’ and ‘citation_sum’, respectively. The online dialog for standard geographical scatter of citations is in Figure 4.11b.



Journal International Impact

Please, enter parameters!

Select journal names (or any):

any
Information Processing and Manag
Information Systems
JASIS, Journal of the American Soci
Journal of Information Science
The Canadian Journal of Informatio

Citation window years:

Start year: 1980 End year: 1997

Fig. 4.11 (b) The online dialog for standard geographical scatter of citations to journals

The result is given in Figure 4.11c. For example, JASIS has received one citation from USA and four from Canada and Finland.

```
{(Information Processing and Manageme...,
        {(Finland, 3)}),
 (Information Systems,
        {(Canada, 2),
         (Finland, 3)}),
 (JASIS, Journal of the American Soci...,
        {(USA, 1),
         (Canada, 4),
         (Finland, 4)}),
 (Journal of Information Science,
        {(Finland, 1)}),
 (The Canadian Journal of Information...,
        {(Finland, 1),
         (Canada, 1)})}
```

Fig. 4.11 (c) Sample result on geographical scatter of citations

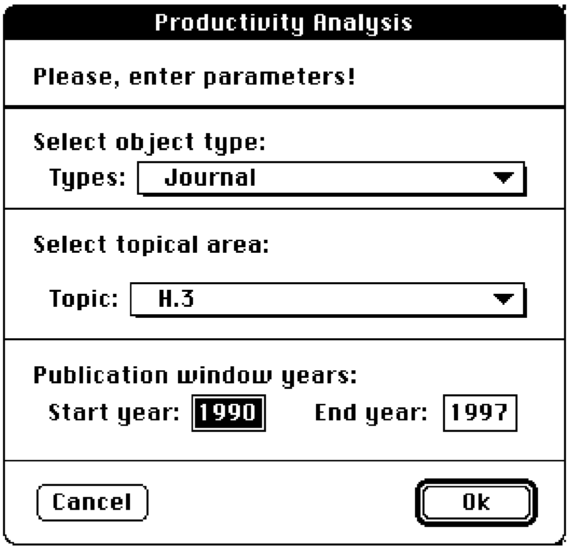
Again, it is straightforward to obtain similar result for other objects of interest, e.g., authors, institutions or countries, by very simple modifications in the **form** constructs which are easily implemented as online dialogs. Current online methods require treatment journal by journal or author by author (Hjortgaard Christensen & Ingwersen, 1996; Ingwersen & Hjortgaard Christensen, 1997). In the FUN interface, again, the statistics are obtained for all relevant journals at once.

4.5. Productivity calculations

The productivity data of journals in a given topical area form the basic data for Bradford's law (e.g., Drott, 1981). The journal productivity figures can be computed by Sample Expression 6 (Figure 4.12a-b). Through the two pop-up menus the user may select among object types author (cf. Lotka's law on publication productivity per scientist), journal, institution and country, and among ACM CS Classes as domains of productivity. However, any available classification or thesaurus may easily be integrated — even several alternative ones, if desired. The publication window may be selected through the edit fields as a range of years. In this case we consider *journal productivity* for articles belonging to the study of "information retrieval" (ACM CS Class 'H.3') published in 1990-97. Ideally, the productivity result should display a Bradford-like distribution.

Sample Expression 6	
form	JOURNAL(journal, art_cnt, IS_ARTS(ano, CLASSES(class)))
relations	ARTICLES
conditions	class **in**(IR_CLASSES(ir_class)) **and** year = **between**([1990, 1997])
aggregation	art_cnt = **cnt**(ano);
subquery	IR_CLASSES(ir_class) = successors('H.3', [SUBCLASS])
sorting	art_cnt
printing	JOURNAL

Fig. 4.12 (a) Sample Expression 6 for journal productivity



Productivity Analysis

Please, enter parameters!

Select object type:
Types: Journal

Select topical area:
Topic: H.3

Publication window years:
Start year: 1990 End year: 1997

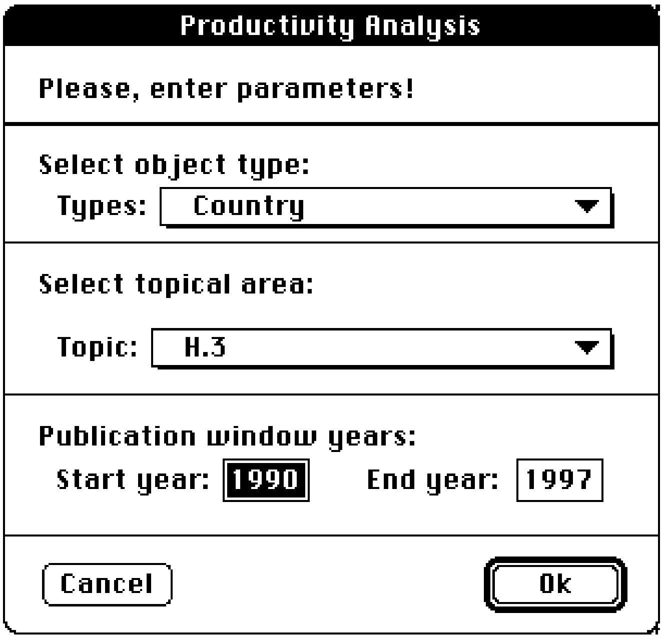
Fig. 4.12 (b) The online dialog for standard journal productivity

The expression constructs a hierarchical relationship between journals and its articles. It also keeps the original relation-valued attribute CLASSES(class) with each article but requires that the classes are classes within the topical area of information systems. The **conditions** and **subquery** components show how transitive relationships in a classification hierarchy are used. Instead of listing all possible subclasses of the class ‘H.3’ (for information retrieval), the user simply asks for all subclasses of ‘H.3’ by the expression successors(‘H.3’, [SUB-CLASS]).

Figure 4.12b presents the online dialog for standard productivity analysis. The object types, for which productivity analysis is available at the moment, are authors (cf. Lotka’s law), institutions, journals and countries. It should be clear that if other object types as analysis units are needed it is very simple to produce the queries — as minor modifications of the **form** construct — and to add the options to the menu. The topical areas are now selected from the ACM CS Classification. However, any classification or thesaurus may easily be integrated — even several alternative ones, if desired. The selections in the dialog of Figure 4.12b correspond to Sample Expression 6 (Figure 4.12a).

```
{(Information Processing and Manageme, 2),
(JASIS, Journal of the American Soci, 3)}
```

Fig. 4.12 (c) Sample result for journal productivity



Productivity Analysis

Please, enter parameters!

Select object type:

Types: Country

Select topical area:

Topic: H.3

Publication window years:

Start year: 1990 End year: 1997

Fig. 4.13 (a) The online dialog for standard country productivity

```
{(USA, 1),
 (Canada, 2),
 (Finland, 2)}
```

Fig. 4.13 (b) Sample result for country productivity

The result data, in Figure 4.12c, report all journals producing articles within the information retrieval area in the 1990's. Thus JASIS has produced 3 articles according to the database. This analysis, focusing on journals, can also be done directly by means of the Dialog RANK command in the ISI databases and is one of the few cases where contemporary online systems are at the level of our approach.

Figure 4.13a-b give the same data organized by countries instead of journals.

4.6. Bibliographic coupling

Two documents are coupled bibliographically when their reference lists have items in common. The number of items in common is called the *coupling frequency*. Bibliographic coupling can be used, e.g., as a retrieval tool for similar (i.e., coupled) articles. (Hjortgaard Christensen & Ingwersen, 1996).

Contemporary methods for online computation of bibliographic coupling are quite complicated. The method described by Hjortgaard Christensen and Ingwersen (1996) consists of a combined multiple step application of the RANK and TARGET commands of the Dialog retrieval system. In our view, the main problem of the method is that it only finds documents coupled with a single source document. Therefore, computing the coupling data for a set of source documents, e.g., representing a research area, first requires identification of the source documents and then using the multiple step RANK and TARGET process for each source document separately. Below we shall show how data on bibliographic coupling can be computed through the FUN interface.

Sample Expression 7 in Figure 4.14a illustrates, how the coupling of articles by a set of authors is computed. We shall consider the authors Crawford, Desai, Lynch and Macleod and their publications in the 1990's. We first compute the auxiliary relation ‘ARTICLES3’ which gives for each author his articles in the 1990's.

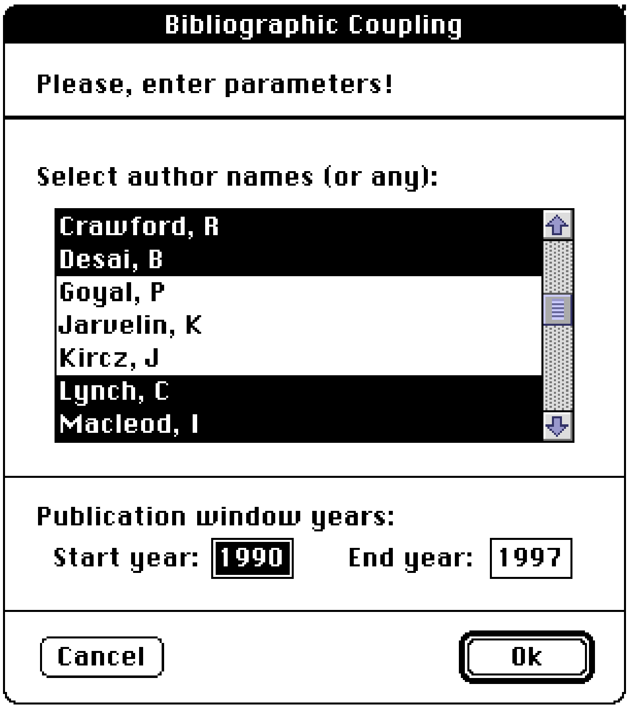
Sample Expression 7

```
ARTICLES3 =
          renaming      {(ano, cc_ano), (author, cc_author)}
          form          ARTICLES2(ano, author)
          relations     ARTICLES
          conditions    year = between([1990, 1997]) and author = in('Crawford, R', … )

form          COUPLINGS(author, cc_author,
                        BIB_COUP(ano, cc_ano, ref_cnt1, ref_cnt2, bibcoco,
                                  REFS1(ref1),
                                  REFS2(ref2),
                                  JOINTREFS(jc_ano)))
relations     ARTICLES, ARTICLES3
conditions    (bibcoco > 0) and (ano ≠ cc_ano) and (author ≠ cc_author) and
              year = between([1990, 1997]) and author = in('Crawford, R', … )
aggregation   bibcoco    = cnt(jc_ano);
              ref_cnt1 = cnt(ref1)
              ref_cnt2 = cnt(ref2)
subquery      REFS1(ref1) = im_successors(cc_ano, [CITES])
              REFS2(ref2) = im_successors(cc_ano, [CITES])
              JOINTREFS(jc_ano) =
                        set_intersection(
                                  im_successors(ano, [CITES]),
                                  im_successors(cc_ano, [CITES]))
sorting       author, bibcoco
printing      COUPLINGS, BIB_COUP
```

Fig. 4.14 (a) Sample Expression 7 for bibliographic coupling of articles

The main expression constructs all author pairs from the two relations ‘ARTICLES’ and ‘ARTICLES2’ on conditions that the articles are published in the 1990's, the authors belong to the required set of four, no-one is coupled with one-self and the coupling coefficient (bibcoco) is greater than zero. Again, we could add conditions on topics, journal, countries, etc., in order to select a subset of authors. For each author pair in the top relation ‘COUPLINGS’, the **form** construct gives article pairs by the two authors and their bibliographic coupling frequency ‘bibcoco’ as well as the reference counts ‘ref_cnt1’ and ‘ref_cnt2’ of the articles given in ‘BIB_COUP’. The references, identified in the attributes ‘ref1’ and ‘ref2’ in the relation-valued attributes ‘REFS1’ and ‘REFS2’, are collected by subqueries in the citation network. This count is used as a normalizer for the coupling frequency. The coupling frequency ‘bibcoco’ is computed from the relation JOINTREFS(jc_ano), which also is computed by a **subquery** giving the intersection of the references of the two articles under consideration.

Because the **conditions** -component does not give an explicit join condition for the two relations, all article pairs are considered. However, all pairs are not reported, because the condition (bibcoco > 0) and (ano ≠ cc_ano) and (author ≠ cc_author) is applied, i.e., two articles need to have a coupling frequency above zero in order to be reported and couplings of one article or author with itself are omitted. Figure 4.14b gives the online dialog for standard bibliographic coupling of authors, corresponding in content to Sample Expression 7.



Bibliographic Coupling

Please, enter parameters!

Select author names (or any):

Crawford, R
Desai, B
Goyal, P
Jarvelin, K
Kircz, J
Lynch, C
Macleod, I

Publication window years:
Start year: 1990 End year: 1997

Fig. 4.14 (b) The online dialog for standard bibliographic coupling of articles

Figure 4.14c shows the result for article coupling frequency. Data are shown only for Macleod and Lynch because the two others did not publish in the 1990's. The data are reported twice because the query does not check for pairs in different order. The data indicate, for example, that the article ‘art_9’ by Lynch is coupled with the article ‘art_10’ by Macleod with the frequency 1. The former has one and the latter two references (recorded in the small database).

```
{(Lynch, C, Macleod, I,
        {(art_9, art_10,  1, 3, 1),
         (art_9,  art_21,    1,  3,
1)}),
 (Macleod, I, Lynch, C,
        {(art_10, art_9,  3, 1, 1),
         (art_21,  art_9,    3,  1,
1)})}
```

Fig. 4.14 (c) Sample result for article bibliographic coupling frequency

Sample Expression 7 computes the article coupling frequencies by author pairs *for all articles* in the set limited by the **conditions** construct. This set can be limited by any conditions on article sources, authors, topics, etc. In other words, a set of articles is processed at once and there is no need to pinpoint each source article one by one in advance. Here we organized the result by author pairs. However, any relevant source (or other derived) attributes may be incorporated into the result.

It is straightforward to continue the analysis by, e.g., normalizing the bibliographic coupling frequencies (bibcoco) by the reference list lengths. It also is possible to aggregate the data across articles to measure the nearness of authors, e.g., by summing the normalized coefficients for each pair of authors.

By extension, one obtains also department (organization, country, journal, ...) coupling data by replacing the author attributes by the attributes representing these other object types. In a very similar way, the user can compute couplings of other data, e.g., broad document classes (here, e.g., the Computer Science classes), keywords (with hierarchies, if needed), disciplines (provided that journals have discipline codes or names). This analytical and computational strength emanates from modeling the document representations as structured objects and from allowing easy declarative restructuring and aggregation.

5. DISCUSSION

We have shown in this paper how informetric calculations can easily and declaratively be specified through advanced data management techniques. We have modeled typical informetric data, bibliographic data, as complex objects (NF^2 relations) as well as terminological and citation networks representing transitive relationships. We have also introduced a very high-level query interface, the FUN interface developed by Niemi and Järvelin (1995; Järvelin & Niemi, 1995; 1997), which enabled us to define ad hoc online queries computing the data for basic informetric concepts. In this paper we developed the FUN interface to offer online dialogs for further simplification of standard informetric analyses.

The benefits of the proposed data modeling and query interface are methodological and conceptual. *Methodologically,* data for basic informetric concepts, such as bibliographic coupling, author co-citation analysis, impact factors, international visibility and international impact, productivity calculations in a given area, etc., can be computed easily and often with much less effort than in contemporary online retrieval systems. More precisely, the methodological benefits can be summarized as follows:

- The user need not process each object of analysis separately as in current online methods (see Hjortgaard Christensen & Ingwersen, 1996; & Wormell, 1997). The objects of analysis can be specified implicitly and declaratively by the **conditions** construct of the FUN query language or through the online dialogs. Explicit identification of relevant objects for the statistics may require considerable experience in the area under consideration.
- The user need not specify each year of citation in citation analyses separately as traditionally (see Hjortgaard Christensen & Ingwersen, 1996). Instead, she gets data for all relevant years automatically.
- Multiple statistics may be computed at once by a single query. For example, one may compute at once for each journal in a research domain the number of articles per journal, the average number of references per article in the journal, and the average number of citations per article in the journal. Current online systems do not support multi-level multi-attribute data aggregation.
- In co-citation and bibliographic coupling analyses, the pairs for which statistics are computed, are formed automatically. In ACA in particular, the user need not create and process each author pair separately as in current online ACA analysis (McCain, 1990).
- New statistically based qualitative data can be computed. For example, the recognized contribution analysis extends citation analysis by reporting qualitative information based on the citing documents as projected by White (1990).

- The user can very easily change focus, the type of objects of interest in the analysis, between articles, journals, authors, departments, organizations or countries. This is done by very simple modifications in the **form** constructs or, at the online dialog level, by selecting the object type in a menu. Breakdowns of data are easily available, e.g., by years or classes, simply by introducing appropriate relation-valued and atomic-valued attributes in the **form** construct. It is equally easy to analyze, e.g., journals by years as it is to analyze years by journals. Thus any object types may form the units of analysis or serve as data breakdown dimensions. In current online systems, such analyses, if at all possible, would require identifying new objects and repeating manually the multiple step process for each (pair) of them (see Hjortgaard Christensen & Ingwersen, 1996; Ingwersen & Hjortgaard Christensen, 1997).
- The user FUN query language is at a very high abstraction level and highly declarative. Therefore the user need not specify explicitly any data restructuring operations. Also the construction of relation-valued attributes based on subqueries is at a very high level. Our idea is that the user describes, very declaratively, only the relationships among the source and result data. In contemporary online retrieval systems often a very low-abstraction level step-by-step procedure is required, whereas in many advanced database systems the skill requirements on behalf of the user are too demanding (see Niemi & Järvelin, 1995). The online dialogs relieve the users from the burden of using the query language at all.

The benefits of the FUN-interface for informetrics are based on data modeling and the interface’s general expressive power. The modeling of bibliographic data as complex objects, which explicitly specify atomic-valued attributes and relation-valued attributes, supports analysis and aggregation of all structural components. The modeling of thesauri and classifications as binary relations supports transitive processing, e.g., automatic query expansion to broad topical areas. The modeling of citations as binary relations supports easy processing both toward cited documents and toward citing documents.

The FUN interface provides a general expressive power allowing data restructuring, aggregation, retrieval, and transitive processing declaratively at a high abstraction level (see Järvelin & Niemi, 1995; 1997). There are no limitations on the organization of the result object types from the available source relation attributes and derived attributes. By placing source relation attributes and derived attributes in suitably arranged relation-valued attributes, complex result objects can be organized and subdivided flexibly. This supports generalized informetrics. The FUN interface as such is a general purpose interface which may be applied also in many areas outside informetrics.

Conceptually, the interface also supports several fruitful generalizations of typical informetric measurements. These are based on substituting traditional foci of analysis, e.g., journals, by other object types such as authors, organizations, countries or classes of a classification. We have shown through sample expressions how impact factors, co-citation frequencies, internationalization statistics, productivity, as well as bibliographic coupling may be generalized from their traditional object types of analysis to any of the object types of journals, articles, authors, departments, organizations, countries, classes, or years. Both diachronic as well as synchronic analyses can be performed easily. These may be accompanied with statistical breakdowns based on any of the remaining object types. We believe that such analyses are needed in generalized informetrics. Moreover, the proposed interface improves, as a spin-off effect, the possibilities of utilizing citation data in IR, following the overlap investigations by McCain (1989) and Pao (1994) within the cognitive framework (Ingwersen, 1996).

Despite of the many benefits, there are several *limitations and issues* which deserve attention. Although the FUN interface has been implemented in Prolog and runs on several platforms (PCs, Macintoshes and Unix machines), it still is rather *a computational prototype* than an full fledged software product. A product would require further developments in efficiency for large amounts of data, user interfaces, and concurrency support. Therefore the data modeling and the FUN interface presented in this paper point out directions how online informetrics may be developed and how this depends on data management techniques. As the interface is now, it would require downloading of data from ISI and other online databases, and automatic conversion to the NF^2 relation representation (which can be done automatically if the source data are of excellent quality). The ISI records should be linked to records from other online databases to complete the citation data by full bibliographic data.

Although the FUN interface provides very high-level declarative *queries,* these *are not always simple* and might require considerable thought on behalf of the user. However, this problem was removed by storing predefined and parameterized queries for use through simple online dialogs. Many of the sample expressions of Section 4 and the Appendix may be expressed in more than one way. Many expressions for auxiliary relations can be avoided by introducing tuple variables (Ullman, 1988) for multiple accesses to the same relation. However, it is clearly easier for the user to use the dialogs and to let a database administrator define the query expressions.

Data quality in source databases is a problem for all informetric analyses (Hjortgaard Christensen & Ingwersen, 1996; Ingwersen & Hjortgaard Christensen, 1997). They have pointed out several problems in online data set creation for informetric analysis:

- structural consistency of items within each database and between databases,
- availability of sufficient data in existing fields,
- consistency of coding of structural components (i.e., field tags),
- consistency of data item representation — e.g., how many different forms there are for person, journal or corporate names,
- consistency and quality of indexing and/or classification.

Lack of consistency and quality in these areas cause problems in data conversion from online databases to the NF^2 relation format of the FUN interface.

6. CONCLUSION

This article demonstrates how informetric calculations can be performed through moderns database management techniques. The article is based on a small sample database and development of sample queries for informetric calculations. The queries are run through the FUN interface that is a computational prototype, which has been implemented in Prolog and runs on several platforms (PCs, Macintoshes and Unix machines). Therefore the data modeling and the FUN interface presented in this paper point out directions how online informetrics may be developed and how this depends on data management techniques.

The contribution demonstrates methodological as well as conceptual benefits for informetrics. First, they can be achieved through advanced data modeling of complex objects as well as terminological and citation networks, and secondly, through high-level declarative query interfaces providing a general expressive power allowing data restructuring, aggregation, retrieval, and transitive processing. In this way data for basic informetric concepts, such as bibliographic coupling, author co-citation analysis, impact factors, international visibility and international impact, productivity calculations in a given area, can be computed easily and often with much less effort than in contemporary online retrieval systems. Simultaneously, basic informetric concepts can also be generalized by substituting traditional foci of analysis, e.g., journals, by other object types, such as authors, organizations, countries or classes of a classification. There are no limitations on the organization of the result object types from the available source relation attributes and derived attributes. Statistical analyses for any of the object types may be refined by breakdowns based on any of the remaining object types. We believe that such analyses foster generalized informetrics.

REFERENCES

Agrawal, R. & Borgida, A. & Jagadish, H.V. (1989). Efficient management of transitive relationships in large data and knowledge bases. In: Clifford, J. & al (Eds.), *The ACM Sigmod Conference,* Portland, Oregon, May 31-June 2, 1989. New York, NY: ACM Press, 1989, pp. 253-262.

Almind, T. & Ingwersen, P. (1997). Informetric analyses on the World Wide Web: Methodological approaches to “webometrics”. *Journal of Documentation, 53*(4): 404-426.

Desai, B.C. & Goyal, P. & Sadri, F. (1987). Non-first normal form universal relations: An application to information retrieval systems. *Information Systems 12*(1): 49 - 55.

Deux, O. (1990). The story of O_2. *IEEE Transactions on Knowledge and Data Engineering, 2*(1): 91-108.

Dialog (1993). Get results with the Dialog RANK command. *Dialog Chronolog 21*(1): 27-33.

Drott, P. (1981). Bradford's law: Theory, empiricism and gaps between. *Library Trends, 30*(1): 41-52. (Special issue on bibliometrics, Summer 1981).

Egghe, L. & Rousseau, R. (1990). Introduction to Informetrics: Quantitative methods in Library, Documentation and Information Science. Amsterdam: Elsevier.

Hjortgaard Christensen, F. & Ingwersen, P. & Wormell, I. (1997). Online determination of the journal impact factor and its international properties. *Scientometrics, 40*(3): 529-540.

Hjortgaard Christensen, F. & Ingwersen, P. (1996). Online citation analysis: a methodological approach. *Scientometrics, 37*(1): 39-62.

Ingwersen, P. (1984). A cognitive view of three selected online search facilities. *Online Review, 8*(5): 465-492.

Ingwersen, P. (1996). Cognitive perspectives of information retrieval interaction: Elements of a cognitive IR theory. *Journal of Documentation, 52*(1): 3-50.

Ingwersen, P. & Hjortgaard Christensen, F. (1997). Data set isolation for bibliometric online analyses of research publications: Fundamental methodological issues. *Journal of the American Society for Information Science, 48*(3): 205-217.

Järvelin, K. & Niemi, T. (1993). Deductive information retrieval based on classifications. *Journal of the American Society for Information Science, 44*(10): 557-578.

Järvelin, K. & Niemi, T. (1995). An NF^2 relational interface for document retrieval, restructuring and aggregation. In: Fox, E. & Ingwersen, P. & Fidel, R. (Eds.), *The 18th International Conference on Research and Development in Information Re-*

trieval (ACM SIGIR '95), Seattle, Wa, July 9-12, 1995. New York, NY: ACM, 1995, pp. 102-110.

Järvelin, K. & Niemi, T. (1997). *Integration of complex objects and transitive relationships for information retrieval.* Tampere: University of Tampere, Department of Computer Science, Report A-1997-11. (accepted to Information Processing & Management, 1999).

Kekäläinen, J. & Järvelin, K. (1998). The impact of query structure and query expansion on retrieval performance. In: Croft, W. B. & Moffat, A. & van Rijsbergen, C.J. & Wilkinson, R. & Zobel, J. (Eds.), *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (ACM SIGIR '98),* Melbourne, Australia, August 24-28, 1998. New York, NY: ACM Press, pp. 130-137.

Library Trends (1981). Special issue on bibliometrics, summer 1981. *Library Trends, 30*(1).

Macleod, I. A. (1990). Storage and retrieval of structured documents. *Information Processing and Management, 26*(2): 197-208.

McCain, K.W. (1989). Descriptor and citation retrieval in the medicine behavioural sciences literature: Retrieval overlaps and novelty. *Journal of American Society for Information Science, 40*: 110-114.

McCain, K.W. (1990). Mapping authors in intellectual space: a technical overview. *Journal of the American Society of Information Science, 41*(6): 433-443.

Moed, H.F. & van Leeuwen, Th.N. (1996). Impact factors can mislead. *Nature, 381*: 186.

Niemi, T. & Järvelin, K. (1992). Operation-Oriented Query Language Approach for Recursive Queries – Part 2. Prototype Implementation and Its Integration with Relational Databases. *Information Systems, 17*(1): 77-106.

Niemi, T. & Järvelin, K. (1995). A Straightforward NF^2 relational interface with applications in information retrieval. *Information Processing & Management, 31*(2): 215-231.

Niemi, T. & Järvelin, K. (1996). The processing strategy for the NF^2 relational FRC-interface. *Information & Software Technology 38:* 11-24.

Pao, M.L. (1994). Relevance odds of retrieval overlaps from seven search fields. *Information Processing & Management, 30*(3)305-314.

Paredaens, J. & Peelman, P. & Tanca, L. (1995). G-log: A Graph-based query language. *IEEE Transactions on Knowledge and Data Engineering, 7*(3): 25-43.

Persson, O. (1999). BibExcel. At: http://www.umu.se/inforsk/. Visited March 10,1999.

Pistor, P. & Andersen F. (1986). Designing a generalized NF^2 model with an sql-type language interface. In: Chu W. et al (Eds.), *The 12th VLDB Conference,* Kyoto, Japan, Aug. 21-23, 1986. Los Altos, CA: Morgan Kaufman, pp. 278-285.

Rada, R. & Wang, W. & Birchall, A. (1993). Retrieval hierarchies in hypertext. *Information Processing & Management, 29*(3): 356-371.

Roth, M.A. & Korth, H.F. & Batory, D.S. (1987). SQL/NF: a query language for ¬1NF relational databases. *Information Systems 12*(1), 99-114.

Salminen, A. & Tague-Sutcliffe, J. & McClellan, C. (1995). From text to hypertext by indexing. *ACM Transactions on Information Systems, 13*(1): 69-99.

Sheck, H.-J. & Scholl, M.H. (1986). The relational model with relation-valued attributes. *Information Systems 11*(2), 137-147.

Smith, J. & Smith, D. (1977). Database abstractions: Aggregation and generalization. *ACM Transactions on Database Systems, 2*(2): 105-133.

Südkamp, N. & Linnemann, V. (1990). Elimination of views and redundant variables in an SQL-like database language for extended NF^2 structures. In: D. McLeod & al. (Ed.), *Proceedings of the 16th VLDB Conference* (pp. 302-313). Palo Alto, CA: Morgan Kaufman Publishers.

Ullman, J.D. (1988). *Principles of database and knowledge base systems. Vol. I.* Rockville, MD: Computer Science Press.

Ullman, J.D. (1989). *Principles of database and knowledge base systems.* Vol. II. Rockville, MD: Computer Science Press.

White, H.D. ed., (1990). Perspectives on author co-citation analysis. *Journal of the American Society of Information Science, 41*(6): 430-468.

White, H.D. & McCain, K.W. (1998). Visualizing a discipline: An author co-citation analysis of information Science, 1972-95. *Journal of American Society for Information Science, 49*(4): 327-355.

Wormell, I. (1998). Informetric analysis of the international impact of scientific journals: How international are the international journals? *Journal of Documentation, 54*(5): 584-605.

APPENDIX: QUERY LANGUAGE SYNTAX

Notations:

- repetition 1 - n times is denoted by $^+$
- repetition 0 - n times is denoted by *
- an optional component is denoted by ^{0} (0 - 1 times repetition)

The production <general_expression> also allows relational algebra (RA) expressions. These, however, are not shown in the grammar, because RA expressions are not used in this paper. The implemented system accepts RA expressions (Järvelin & Niemi, 1997).

```
<FUNQuery>        →      form <form>
```

relations <relation_name>$^+$

conditions <condition>

aggregation <aggregation_defn>*

deduction <expression_declaration>*

sorting <attribute_name>*

printing <relation_name>$^+$

<form> → <relation_name>(<attribute_name>$^+$ {, <form>}^{0})

<relation_name> → <word>

<condition> → <factor> {**or** <factor>}*

<factor> → {**neg**}^{0} <logical_element> {**and** <factor>}*

<logical_element> → <av_condition> | <aa_condition> | <in_condition> |
<between_condition> | <group_condition> | <match_condition>
| **(**<condition>**)**

<av_condition> → <attribute_name> <CompOp> <string>

<aa_condition> → <attribute_name> <CompOp> <attribute_name>

<in_condition> → <attribute_name> **in(**<value_set>**)** |
<attribute_name> **in(**<form>**)**

<between_condition> → <attribute_name> **between(**<numeric>, <numeric>**)**

<group_condition> → **group(**<relation_name>**,** <condition>**)**

<match_condition> → <attribute_name> **match(**<text_tag>**,** <text_pattern>**)**

<text_pattern> → <text_factor> {**or** <text_factor>}*

<text_factor> → <text_element> {**and** <text_factor>}*

<text_element> → <proxpattern> | <pattern> | (<text_pattern>) |
neg <text_pattern>

<proxpattern> → **prox(**<distance>, <pattern>, <pattern>**)** | % ordered proximity
adj(<distance>, <pattern>, <pattern>**)** % unordered adjacency

<pattern> → **st(**<string>, <masklenght>**)** | % wild character 0 - n
st1(<string>**)** | % wild character of lenght 1
bw(<string>**)** | % whole word matching

<distance> → <integer>

<masklenght> → <integer>

<string> → <word> | <integer> | <code>

<word> → <letter> | <letter><word>

<numeric> → <integer> | <real>

<integer> → <number> | <number><integer>

<real> → <integer>.<integer>

<code> → <word><integer> | <integer><word>

<letter> → "A" | "B" | ... | "Ä" | "Å" | "a" | "b" | ... | "ä" | "å"

<number> → "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" | "0"

<character> → <letter> | <number>

<delimiter> → " " | "." | "," | ":" | ";" | "!" | "?" | "&" | """" | "#" | "£" | "$" | "¢" |
"%" | "‰" | "§" | "¶" | "(" | ")" | "[" | "]" | "{" | "}" | "+" | "-" | "/"
| "*" | """ | "_" | "<" | "≤" | "=" | "≥" | ">" | "≠"

<sentence delimiter> → "." | "!" | "?" | "¶"

<text_tag> → <word>

<aggregation_defn> → <attribute_name> <aggregation_op> <attribute_name>

<aggregation_op> → **cnt** | **avg** | **min** | **max** | **sum**

<expression_declaration> → <form> **=** <general_expression>

<attribute_name> → <word>

<value_set> → **{**<string>{**,** <string>}*****}**

<general_ expression> → <FUNQuery> | <transitive_expression> |
<renamed_expression>

<transitive_expression> → <node-operation>

<node-operation> → <node_set>

<node-operation> → **ROOT_NODES(**<scope>**)** | **LEAF_NODES(**<scope>**)**

<node-operation> → **NODES(**<scope>**)**

<node-operation> → **SET_INTERSECTION(**<node-operation>**,** <node-operation>**)**

<node-operation> → **SET_DIFFERENCE(**<node-operation>**,** <node-operation>**)**

<node-operation> → **SET_UNION(**<node-operation>**,** <node-operation>**)**

<node-operation> → **IM_SUCCESSORS(**<node>**,** <scope>**)**

<node-operation> → **IM_PREDECESSORS(**<node>**,** <scope>**)**

<node-operation> → **SUCCESSORS(**<node>**,** <scope>**)**

<node-operation> → **PREDECESSORS(**<node>**,** <scope>**)**

<node-operation> → **UNION_OF_SUCCESSORS(**<node-operation>**,** <scope>**)**

<node-operation> → **UNION_OF_PREDECESSORS(**<node-operation>**,** <scope>**)**

<node-operation> → **INTERSECTION_OF_SUCCESSORS(**<node-operation>**,** <scope>**)**

<node-operation> → **INTERSECTION_OF_PREDECESSORS(**<node-operation>**,** <scope>**)**

<node-operation> → **DIFFERENCE_OF_SUCCESSORS(**<node-operation>**,** <node-operation>**,** <scope>**)**

<node-operation> → **DIFFERENCE_OF_PREDECESSORS(**<node-operation>**,** <node-operation>**,** <scope>**)**

<node-set> → **{**<node>{**,** <node>}***}**

<node> → <word> | <code>

<scope> → **{**<transitive_rel_name> {, <transitive_rel_name>}***}**

<renamed_expression> → **RENAMING(**<mapping>, <FUNQuery>**)**

<renamed_expression> → **RENAMING(**<mapping>, <transitive_expression>**)**

<mapping> → **{(**<attribute_name>, <attribute_name>**)** {, **(**<attribute_name>, <attribute_name>**)**}***}**