The DBLP Computer Science Bibliography: Evolution, Research Issues, Perspectives

Michael Ley

University of Trier, FB 4 – Informatik
D-54286 Trier, Germany
ley@uni-trier.de

Abstract. Publications are essential for scientific communication. Access to publications is provided by conventional libraries, digital libraries operated by learned societies or commercial publishers, and a huge number of web sites maintained by the scientists themselves or their institutions. Comprehensive meta-indices for this increasing number of information sources are missing for most areas of science. The DBLP Computer Science Bibliography of the University of Trier has grown from a very specialized small collection of bibliographic information to a major part of the infrastructure used by thousands of computer scientists. This short paper first reports the history of DBLP and sketches the very simple software behind the service. The most time-consuming task for the maintainers of DBLP may be viewed as a special instance of the authority control problem: how to normalize different spellings of person names. The third section of the paper discusses some details of this problem which might be an interesting research issue for the information retrieval community.

1 History

1.1 The Beginning

DBLP was started at the end of 1993 as a simple test of Web technology which became popular with the Xmosaic browsers and the NCSA HTTP server in this year. A byproduct of the author's just finalized PhD thesis was a collection of photocopied tables of contents (TOCs) of important proceedings and journals from the fields of database systems and logic programming. These TOCs were typed in and formatted with some basic HTML markup, and a few entry pages were added. The server was called "Data Bases and Logic Programming" (DBLP) and announced in the dbworld mail list. Surprisingly this very primitive Web server seemed to be useful for others.

A.H.F. Laender and A.L. Oliveira (Eds.): SPIRE 2002, LNCS 2476, pp. 1-10, 2002.

1.2 Two Very Early Decisions

It is obvious that a simple hierarchy of text pages neither explores the expressibility of the medium hypertext nor constitutes an adequate interface for a bibliography. By adding a hyperlink from each author's name to a page which enumerates this person's publications a network orthogonal to the TOC hierarchy was introduced. The author pages contain links to coauthors and to the corresponding TOC pages. To make this "person-publication network" [L] browsable makes sense, because it is a result of the social network behind research. A click to a person page may answer questions like "Did the author publish other papers about this subject/project?", "In which journals/conferences did the author publish?", "When did the author start publishing?", etc. An important implication of the introduction of author pages is that the spelling of person names has to be normalized to get a 1:1 mapping between persons and person pages whenever possible. The person name normalization now is the hardest problem for the maintenance of DBLP. We will discuss it in the third section.

A second and perhaps unusual decision in an academic environment was that DBLP is a service and not a research project. The service is operated with very limited resources. There is always a trade-off between the development of new features or supporting software and entering or maintaining contents on a satisfactory level of quality. Most time we decided in favor of contents. This was supported by the experience that maintenance of complex software systems may cost more time than the gain in productivity justifies for our (almost) single user mode of operation. The authoritative version of DBLP has not used a database management system until today. In 1996 a student developed software to operate DBLP with the object-oriented DBMS SHORE from the University of Wisconsin. Two years later another student repeated the experiment with DB2 from IBM. Both diploma theses resulted in operational software, but it was too time consuming to keep them running on changing machines and operating system versions of the Unix/Linux family. For the production of DBLP a small set of programs and scripts written in C, Perl, Shell, and Java is still sufficient. The only non-standard software package we use is the MG information retrieval system described in the excellent "Managing Gigabytes" book [WMB].

1.3 Early Recognition

In 1997 we received the ACM SIGMOD Service Award and a VLDB Endowment Special Recognition Award. These awards by leading institutions of the database research field helped to give DBLP a more official status at the department of computer science and to get a small initial fund which enabled us to hire a student for entering data into DBLP for a year.

1.4 SIGMOD Anthology

In November 1997 we were contacted by Rick Snodgrass, chair of ACM SIGMOD. With its conferences SIGMOD had made some profit. Rick's idea was to use the money to scan in as much as possible of the "historical" publications of the database

field and to combine the papers with an enhanced version of DBLP to a CDROM archive. Most money was spent to scan in the material and to convert it to PDF; this was done by Pinehurst Inc. In Trier we hired students from SIGMOD funds to improve the coverage of DBLP for the database field and for many boring editorial tasks during the project. To get a better idea of what is missing in DBLP and/or the SIGMOD Anthology we entered more than 100000 citation links for mainstream database publications. Many database publications are now annotated with a (partial) "referenced by" list of incoming citations.

The first 5 CDROMs of the Anthology were distributed in 1999 to the SIGMOD members. After this the project was joined by more and more conference organizers and journal editors which wanted to include their publications into the CDROM collection. Until the end of 2001 we produced 21 CDROMs with more than 150000 pages of material. Finally all material including a snapshot of DBLP from January 2002 was composed on two DVDs to get a very compact digital library containing most important research publications from the database field written before 2000. The realization of this huge collection was only possible with the help of more than 60 people which contributed material and permissions from copyright holders. The excellent management by Rick Snodgrass was essential to make it happen.

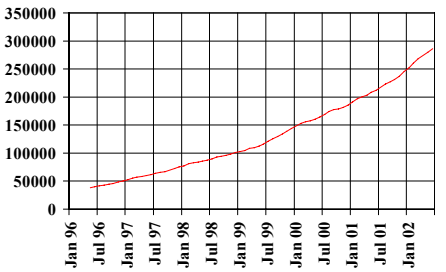

Fig. 1. Growth of DBLP

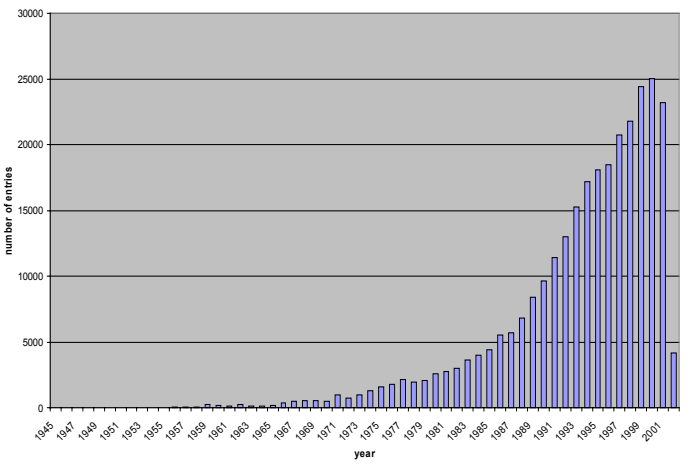

Fig. 2. DBLP entries by publication year

1.5 Sponsor Found

The Anthology project helped to make DBLP the most comprehensive bibliography of the database field. This field is a central part of computer science and has strong ties to other fields like operating systems, information retrieval, programming languages, algorithms, complexity theory etc. The goal is to extend the coverage of DBLP to all areas of computer science. With the generous support of the Microsoft Bay Area Research Center and Jim Gray we were able to hire students which work toward this direction. For the non-database fields DBLP is far from being "complete", but feedback from users indicates that it is becoming useful for researchers from a wide range of fields. As of now (June 2002), the bibliography lists more than 286000 publications. For more than 76000 of them it provides direct links to the abstracts and/or full texts stored on the publishers Web sites. Fig.1 shows the growth of DBLP and Fig. 2 shows the number of entries by publication year.

1.6 Undergraduate Software Labs

For CS students DBLP may be a convenient access path to primary scientific publications. But there is another important usage of the DBLP data set: At the University of Trier we teach Java as the first programming language. Students have to write programs which analyze the structure of DBLP, for example by applying classical graph algorithms to the coauthor relationship or by generating visualizations of simple statistics. The XML representation of DBLP is a medium size (>100MB) 'real' data set. To process it efficiently, careful selection of algorithms and clean design of programs is necessary. The number of temporary objects constructed often becomes the most time-critical parameter of student programs. DBLP data are used as a test set by several researchers, for example [LS], [PG].

2 Technical Background

DBLP is operated with very simple software. In this section we explain how it works.

2.1 HTML TOCs

The initial collection of HTML tables of contents had a very homogenous structure. A TOC is an enumeration of article level bibliographic information. Session titles, volume level bibliographic information, or cross-references may be added. The article level information was included in "unordered lists" (<ul> elements), anything else was outside of <ul>. To generate author pages we took a two step approach:

(1) A customized version of the HTML parser from the xmosaic browser was used to parse the TOC files. The parser was started from a shell script for each TOC. Volume level information like conference names or journal names was hard coded in the shell-script. The result was a large line oriented refer-style like file which contained all information required to generate the author pages ("TOC_OUT").

(2) A second program ("mkauthors") reads this file into a main memory data structure and generates all author pages. We considered computing author pages on demand, but we decided that it is simpler and cheaper to provide the disk space for the materialized pages than to generate an index and to compose each page when required. mkauthors runs every evening if any DBLP data has been changed.

2.2 Mirrors

When DBLP was started, the University of Trier only had a slow connection to the Internet. The reliability of this connection and our computing environment was poor. To make DBLP a useful service, it should have a high availability. To achieve this, we decided to establish mirrors of the server. The first one was implemented on the SunSite Central Europe host operated by the group of Matthias Jarke at Aachen University of Technology. The technique of mirroring DBLP is primitive: The TOC_OUT file and all HTML pages except the author pages are packed into a compressed tar-file. The transfer of this archive is triggered manually or by a simple demon process on the mirror host. After unpacking mkauthor is run on the mirror.

2.3 Simple Search

Two very simple search programs were added to DBLP: The author search program is a sequential substring matching algorithm which runs through a file with all author names. This file is produced as a side effect by mkauthors. Author search is insensitive to umlauts and accents. The title search directly operates on the TOC_OUT file. Both search engines are C programs triggered by the HTTP demon via the CGI interface. The sequential search never was a serious performance problem, but substring matching author or title words without (Boolean) operators, stemming, ranking etc. is an insufficient level of service. The important advantage of this primitive technique is that it never was a problem to install it on mirror hosts.

2.4 XML Records

It soon became apparent that it is not a good design to use the HTML formatted tables of contents as the primary location to store all bibliographic information. Citation linking, annotated bibliographies, reading lists etc. make it necessary to assign a unique ID to each publication and to store the information in more classical bibliographic records. The standard format for bibliographic records in computer science is BibTeX. Because BibTeX is not easy to parse, we decided to use HTML-style markup but BibTeX publication types and field names. Again we customized the xmosaic parser. Later we noticed that our records perfectly fit into the new XML framework, we only had to adapt some minor syntactic details. Fig. 3 shows an example of a DBLP record. On the primary host each record is stored in a separate file; the key is interpreted as a file name.

```
<article key="journals/algorithmica/NavarroB01">
  <author>Gonzalo Navarro</author>
  <author>Ricardo A. Baeza-Yates</author>
  <title>Improving an Algorithm for Approximate
       Pattern Matching.</title>
  <pages>473-502</pages>
  <year>2001</year>
  <volume>30</volume>
  <journal>Algorithmica</journal>
  <number>4</number>
  <ee>http://link.springer.de/link/service/journals/00453/
    contents/01/0034/</ee>
  <url>db/journals/algorithmica/algorithmica30.html#NavarroB01</url>
</article>
```

Fig. 3. A DBLP XML Record

2.5 BHT (Bibliography HyperText)

Not all information from the tables of content was moved to the XML records. Session titles, cross-references between HTML pages and many unstructured comments and annotations can not be represented in BibTeX style records without lots of new but rarely used fields. We decided to leave this information in the tables of content. The XML records are imported into the tables of contents by a simple include mechanism which is implemented by a macro-processor: 'mkhtml' reads a 'bibliography hypertext' (BHT) file and replaces each <cite key="..." style="..."> by the XML record with the specified key using the format indicated by the style attribute. Additionally mkhtml knows tags to generate logos and page footers with navigation bars. Again all HTML pages are produced in advance. This requires more disk space than an 'on demand' strategy, but it requires less disk bandwidth.

2.6 MG

The XML records are loaded into the MG information retrieval system. A CGI based Web interface represents query results of this 'advanced search' engine in a style similar to the author pages. The MG backend treats each XML record as a small text document, only the front end is able to interpret the field structure of the records for filtering out 'false drops' caused by hits in fields different from the query specification.

During the ACM SIGMOD Anthology project students have been entering more than 100000 citation links into DBLP. Each reference from selected papers had to be located in the DBLP collection. The keys of hits are stored in <cite>-fields added to the XML-records of the starting nodes of the citation arcs. The plain text based MG system without the Web front end proved to be a superior tool for this highly repetitive task after a short time of familiarization.

2.7 Anthology Full Text Search

Another kind of information retrieval engine is employed on the CDROMs/DVDs of the SIGMOD Anthology: With Acrobat it is possible to build full text indices of collections of PDF documents. The Acrobat Reader version “with search” uses these indices for efficient full text search on complete CDROMs or DVDs. A prerequisite for indexing scanned material is to run all documents through an OCR algorithm, which is available as a part of the Acrobat software.

3 Research Issues

3.1 Person Names

Person names are ubiquitous in information systems. They are the only widely accepted method to identify persons. For humans the use of familiar person names is superior to all artificial identifiers. For information systems person names have many drawbacks compared to synthetically generated identifiers. Person names may not be unique; several persons with the same name may exist. A person may change his or her name; often several variations of a person name are used.

Variations of names may be caused by abbreviations (for example Jeffrey D. Ullman may become J. D. Ullman, J. Ullman or Jeff Ullman), nicknames (e.g. Michael may become Mike, William / Bill, Joseph / Joe etc.), permutations (e.g. Liu Bin may be the same person as Bin Liu), different transcriptions (e.g. Andrei / Andrey / Andrej may be identical), accents (Stephane vs. Stéphane), umlauts (Muller / Müller / Mueller), ligatures (Weiß / Weiss or Åström / Aastrom ...), case (Al-A'Ali vs. Al-A'ali), hyphens (Hans-Peter vs. Hans Peter), composition (MaoLin Qin vs. Mao Lin Qin), postfixes (Jr./Sr., a number, or a parents name), and by typos. Names may be changed at marriage or emigration to another cultural environment.

For DBLP the variations in person names have two consequences: (1) It is unclear how to search persons best, and (2) it may be hard to normalize different spellings of names to get correct author pages.

3.2 Person Search

The DBLP author search engine on the Web uses a simple substring matching algorithm which is insensitive to case, accents, and umlauts. If a known part of the name, usually the last name, is longer than 4 or 5 characters, the algorithm is practicable to locate the correct author page. For very common short names like Kim, Chen, Chan etc. the precision may be very poor.

The CDROM/DVD version of DBLP published as part of the ACM SIGMOD Anthology uses another search program: The Java Applet accepts queries which are sets of space separated strings. Names in the collection are tokenized into name parts. A name matches a query, if all query strings are prefixes of the parts of the name, i.e. the query "J Ullm" finds "J. D. Ullman", "Jeff Ullman", "Jeffrey D. Ullman", etc. The data structure behind the Applet is an ordered list of all permutations of the name

parts (a KWIC). The list is stored in a simple file based tree. The longest query string is used to select a part of the tree; this sub list is filtered according to the other query strings. In most situations this algorithm is superior to the first one.

3.3 Normalization

We ourselves are often very heavy users of the person search programs. For each publication to be entered into DBLP we try to locate the authors (or editors) in the existing collection. If the spellings differ, but we are very confident that they are variations of the same person's name, we make them equal. It may be necessary to change the spelling in the new entry, in old entries, or both. We try to use a name variation preferred by the person which writes out most of the name parts. For persons who authored or edited many publications the name spelling usually converges very fast to a stable and correct state. For persons with a few known publications the likelihood for duplicate author pages, incorrect, or incomplete spellings is much higher.

The decision of when to unify two spellings may be very complex. To increase the productivity of maintaining DBLP (and many other information systems), this step should be supported by specialized tools. Many of the heuristics we use in the decision process are computable, but some of them require massive background knowledge which isn't available to a realistic system.

A very simple observation is that most publications have been written by groups of authors. These groups are part of working groups or collaborations embedded in theme-oriented communities. The coauthor relationship often gives very strong indications for the identity of persons. Conference series or specialized journals are condensation points for communities. These are larger and fuzzier than coauthor groups, but nevertheless it is often very helpful to look at the journal and conference names to get a rating. The focus of research of persons, groups, and communities changes with the time, an important precondition is that the new facts are consistent with the time frame of known facts.

3.4 Rating Functions

Throughout the last years the focus of information retrieval research has moved from large collections of isolated texts to hypermedia. For most search engines the link structure of documents has become a very important for rating functions. Terms inside Web pages are weighted in dependence of their position in the markup tree, e.g. words inside headings usually are considered more important than words inside long paragraphs. Bibliographic records are typical "semi-structured" documents. Information about authors and the coordinates of the publication (journal, volume, page, year, etc.) are database fields with well known semantics. Titles, abstracts, annotations, and reviews are examples of information inside of bibliographic records which may have a more irregular rich structure.

The idea is to develop specialized weight functions for the name normalization problem. The “query” is a set of names, a title, and the coordinates of the new publication. The task is to add the new bibliographic record to the collection in an “intelligent” manner. For DBLP name normalization has high priority, but a positive

side effect of the decision process is that we may find other inconsistencies in the collection.

3.5 DBLP Browser

The ideas on how to support name normalization are at a very early stage. Any system support will require extensive experimentation and evaluation. A very fast interactive environment to explore the DBLP collection is needed. It should provide a convenient graphical user interface with specialized visualization facilities for the DBLP collection.

From an information retrieval viewpoint DBLP is a small collection. As of June 2002 it contains 286000 records, the XML representation of the data set has a size of approx. 115Mbytes. With known data compression techniques it is possible to build a representation which requires <30Mytes. This data structure contains all information from the XML records and is suitable for easy navigation and search within the data set. A Java prototype of the DBLP browser needs only a few seconds to load this compressed representation from a file into the main memory.

The compressed file is a text file, which only contains bytes in the range 0x20-0xff and line breaks. In the main section each bibliographic record is represented by one line. In the file header the total number of bytes and the number of records is specified. The first character of each line is a code for the publication type (article, book, inproceedings, …); the remainder is a sequence of fields. The first byte of each field is a code for the field type (author, title, …); the interpretation of the field value depends on the type.

For titles we use canonical Huffman codes on the word level. The lexicon is stored in a separate section of the text file using the "3-in-4 front-coding" strategy described in the MG book [WMB]. Contrary to MG, we do not use binary Huffman codes. We construct a Huffman tree of degree 213. The digits of the Huffman codes are represented as characters in the range 0x2a-0xff. Title fields are terminated by ')' (=0x29). The use of byte codes makes it very convenient to search on the compressed text [SNZB]. The degree 213 (and not 255) was chosen to make it easy to embed the Huffman coded titles into the byte coded bibliographic records.

Person names in author or editor fields are normalized. Normalization is favorable for a simple compression technique: All person names are stored in a separate section of the file, sorted by frequency of occurrence. For author or editor fields the values are person numbers. The numbers are represented as sequences of base 111 digits. The first digits are bytes in the range 0x21-0x8f the last digit of each number is marked by the transformation to the range 0x90-0xfe.

The same technique is used for other normalized domains: journal names, booktitles (conference names), series names and publisher names are stored in different sections of the file. On the field level only numbers are used.

DBLP records contain paths in several fields. They are keys or URLs. Because many paths share their prefixes, it makes sense to tokenize paths and to store the path elements in an extra section of the file. The path elements in this section are linked bottom-up: Each number stored in a record field points to the rightmost path element. The table entry contains its string representation and the index of the next path element on the left. Roots contain a circular link.

Other domain specific compression techniques proofed to be very efficient: Year values are represented as difference from the maximum year value. The maximum value is stored in the file header. This results in a considerable compression because most DBLP records describe "new" publications [Fig. 2]. Page fields many contain arbitrary strings, but most page values specify intervals of the type "start page – end page". This type of page field values is represented by two base 111 numbers (start page, number of pages). Only page values which are not of this type are represented by strings.

Our intention is to use the DBLP browser as a framework for experiments. It already provides an improved version of the person pages. The compressed version of the DBLP data and the source code of the browser prototype are available on **http://dblp.uni-trier.de/dblpbr/**, we encourage others to use and/or improve it.

4 Perspectives

With better tools it should be possible to improve productivity. But a really comprehensive portal to computer science literature requires more resources and staff. In a very recently approved project we will cooperate with the producer of CompuScience (FIZ Karlsruhe http://www.zblmath.fiz-karlsruhe.de/), the German Computer Society (GI, http://www.gi-ev.de/) and others. The project will be funded by the German Federal Ministry of Education and Research. The objective is to provide an open portal with improved coverage and additional services.

References

[L] Michael Ley: Die Trierer Informatik-Bibliographie DBLP. GI-Jahrestagung 1997: 257-266.

[LS] Hartmut Liefke, Dan Suciu: XMill: an Efficient Compressor for XML Data. SIGMOD Conf. 2000, 153-164.

[PG] Neoklis Polyzotis, Minos N. Garofalakis: Statistical Synopses for Graph-Structured XML Databases. SIGMOD Conf. 2002.

[SNZB] Edleno Silva de Moura, Gonzalo Navarro, Nivio Ziviani, Ricardo Baeza-Yates: Fast and Flexible Word Searching on Compressed Text. TOIS 18(2): 113-139 (2000).

[WMB] Ian H. Witten, Alistair Moffat, Timothy C. Bell: Managing Gigabytes, Compressing and Indexing Documents and Images, 2nd Ed., Morgan Kaufmann, 1999.

Michael Ley was born in Düsseldorf, Germany, in 1959. He received his Diploma in computer science (with honors) from Aachen University of Technology in 1986 and the Ph.D. in computer science (with honors) from the University of Trier in 1993. He is a lecturer at the Department of Computer Science at the University of Trier.