CS315 Project
CSE-IITK Research Paper Database

Abhishek Kar (Y8021)
Aditya Huddedar (Y8221)

Introduction

A bibliographic database is a database of bibliographic records, an organized digital collection of references to published literature, including journal and newspaper articles, conference proceedings, reports, government and legal publications, patents, books, etc. In contrast to library catalog entries, a large proportion of the bibliographic records in bibliographic databases describe analytics (articles, conference papers, etc.) rather than complete monographs, and they generally contain very rich subject descriptions in the form of keywords, subject classification terms, or abstracts.

The aim of our project is to build an easily maintainable and flexible bibliographic database of Research papers published by the faculty of Computer Science and Engineering Department, Indian Institute of Technology Kanpur. The motivation behind our venture was lack of a properly maintained database by the Department. Our database allows efficient searching and convenient display of results. There have been many instances of databases that have attempted to provide easy access to a large corpus of research material (DBLP, CiteSeerx, PubMed). We attempt to build a user friendly interface with a robust query backend that enables accurate search results and easy browsing of search results by providing helpful links and suggested search results. We will rely on indexing the entries with number of citations to provide relevant search results and suggested searches.

Functional Specifications and Features

- **Database**: We created the database from the CSE department publications page. The site is pretty old and contains papers from 2002-2006. We had to manually enter the details for each paper as BibTeX entries would not have sufficed for the information we needed (citations, author details etc.). The author table contains only details of current faculty in the department. We also maintain the broad research area corresponding to each paper. As there were not enough citations between the papers we used, we had to insert dummy citations to test our code.

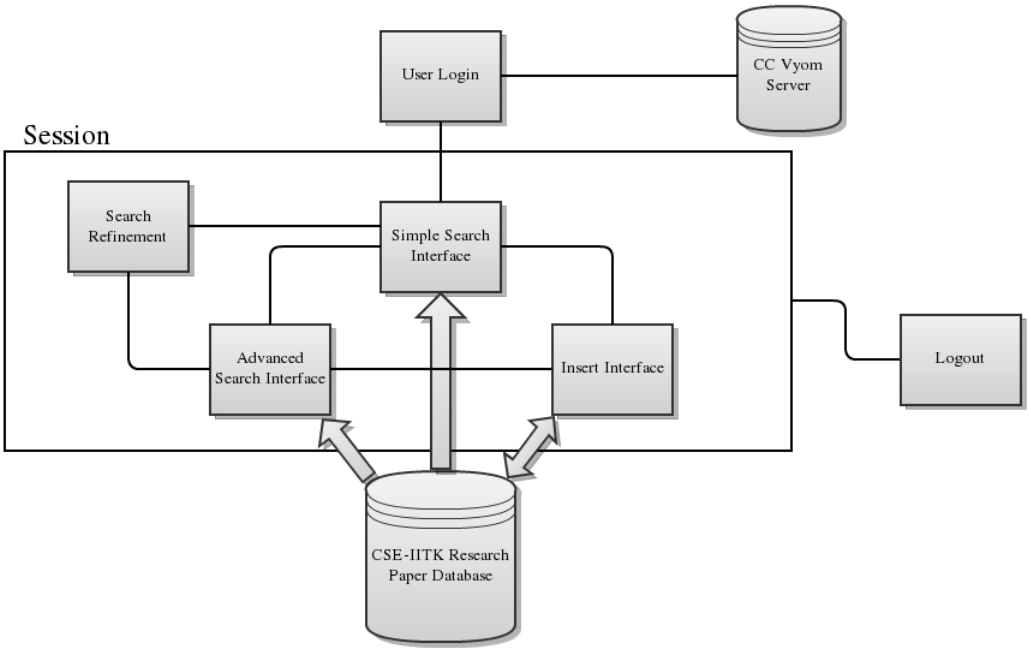
- **Login**:We implemented login to the search service through the IITK Vyom server. It can only be accessed by valid IITK login IDs and passwords. We did this by opening an ftp connection to the Vyom server and authenticating with it using the entered details. We use php sessions to keep track of the user's activites during the session.

- **Session Management**: PHP Sessions are used to pass information from one page to another while a user is logged. Warnings are issued if an already logged in user tries to log in again or another user tries to log in without the first user not being properly logged out.Session variables are also used to grant privileges to certain users to insert into the database.

- **Simple Search**: This interface provides a minimalistic interface for searching the database. We tokenize the input string for various possible delimiters and search in all fields. We display the result sorted by the rank we maintain in the database. The links to the papers are given using the DOI. We also display the author name(s) and publication details. A link is also display mentioning the citations to this paper by clicking on which the papers citing this paper can be viewed.

- **Advanced Search**:The advanced search interface provides a customised search option where a number of fields can be specified and the results are displayed according to it. The result display is the same as above.
- **Result Refinement**:We provide an option to refine the search result by research area and author. The options are display on the search result page and display filtered results.
- **Ranking by Citations**:We maintain a paper rank by the number of papers that cite the paper. The results are displayed according to this rank.
- **Insert into Database**: We provide a user friendly interface to insert data into the database. Javascript is used to check the validity of the fields entered. Insert privileges are only with certain users (currently root users) and it is implemented using PHP session variables. Insertion takes care of new authors and conferences being added to the database and adjusts the entries in the table accordingly.

Flow Chart



Implementation details

We divide the queries into two categories: External and Internal. The external set of queries are the queries that are built based on user input. In the general search option, we search in all fields. In the advanced search option, we provide a form based interface where the SQL query is built based on the user input. The internal queries are for refinement on the search results and for suggested searches.

External:

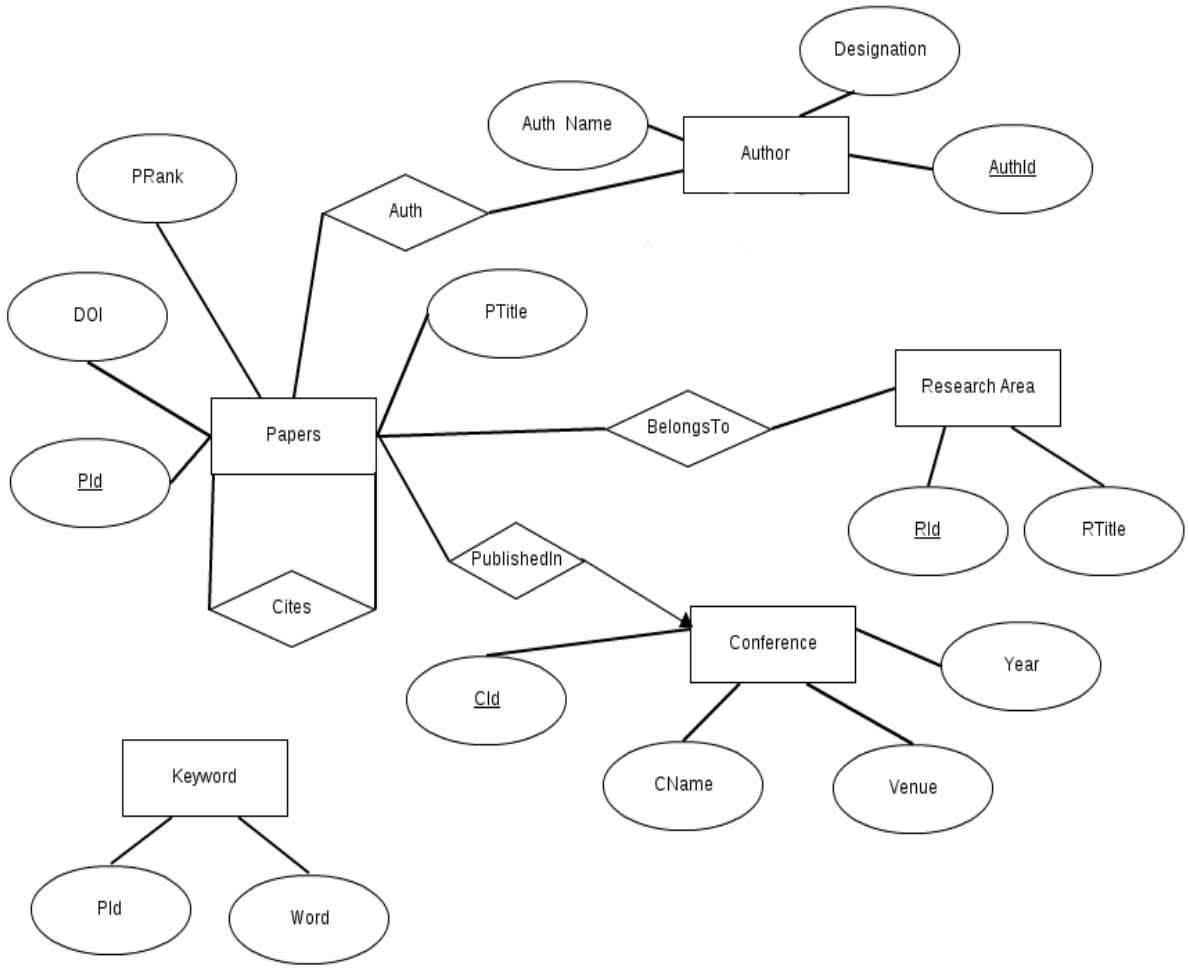
- General Search – Search in all fields
- Search by Title
- Search by Author(s)
- Search by Keyword

- Search by Year
- Search by Conference
- Search by Research Area

Internal:

- Search refinement by Author and Research Area from within search results
- Suggested search: Provide the research papers from the same Research Area (limit 5) + Papers with similar keywords (limit 5)
- Sort results by PRank i.e. Citations.

ER Diagram



Tables

- **EntitySets:**
  1. **(E1)Papers (PId,PTitle,DOI,PRank):**
     Paper table consists of the details of all the research papers in our database. PId is the unique Id which refers to a paper. Hence, it is a key of the Paper table. With each paper, we store its Title(PTitle) and Digital Obejct Identification Number(DOI). DOI can be used later to retrieve the whole paper using http://dx.doi.org. We also rank the papers using the CitedBy count.
  2. **(E2)Author(AuthId, AuthName, Designation):**
     Author table stores the details of all the authors. AuthId is the key of the table which refers to a unique Author.With each author, we store his/her Name and Designation.
  3. **(E3)Conference (CId, CName, Venue, Year):**
     Conference table models a conference. We store a unique CId for each conference. We also store the other details of the conference like Name, Venue and Year.
  4. **(E4)Research Area(RId,RTitle):**
     We identify different research areas to which the papers in our database belong. Based on that we create a table of all the research areas in which each is assigned a unique RId.
  5. **(E5)Keyword (PId, Word):**
     We store the keywords of all the research papers as a table of PId and Word. To avoid redundancy, we do not store it along with the other details of the paper in 'Papers' table.
- **Relationships:**
  1. **(R1)Auth(PId, AuthId):**
     Auth table stores authors for each paper. It is a many-many relationship between Papers and Author.
  2. **(R2)Cites(FromPId, ToPId):**
     Its a many-many relationship between Papers and Papers. FromPId is the PId of a paper which cites a paper with PId ToPId.
  3. **(R3)PublishedIn (PId, CId):**
     This table stores the conference in which the paper was published.Its a many-one relationship from Papers to Conference.
  4. **(R4)BelongsTo (PId, RId):**
     Stores the research area to which the paper belongs.A many-many relationship between Papers and Research Area.

Normalization

We have the following functional dependencies in our database schema.

1. $F1$={PId → PTitle, DOI, PRank} in $E1$ (Key:PId)
2. $F2$={PId → CId} in $R3$ (Key:PId)
3. $F3$={AuthId → AuthName,Designation} in $E2$ (Key:AuthId)
4. $F4$={CId → CName, Venue, Year} in $E3$ (Key:CId)
5. $F5$={RId → RTitle} in $E4$ (Key:RId)

Let us consider the canonical cover of functional dependencies in the whole collection of schema(i.e. before decomposition)
$F = \{$
PId → PTitle, DOI, PRank,
PId → CId ,
AuthId → AuthName, Designation ,
CId → CName, Venue, Year ,
RId → RTitle $\}$

We note that:(Assuming original schema as union of all the table-schema)

1. If we take a join over all the tables,we do not lose any information. Hence, The decomposition is lossless.

2. As all the dependencies are preserved even after decomposition,
   $F^+ = (F1 \cup F2 \cup F3 \cup F4 \cup F5)^+$ (because $F = (F1 \cup F2 \cup F3 \cup F4 \cup F5)$)

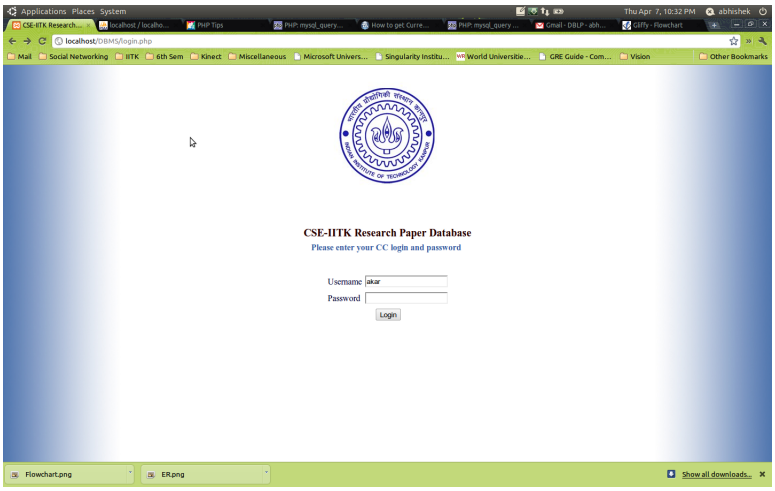
3. We can easily verify that every nontrivial dependency of the form $\alpha \rightarrow \beta$ in each F_i^+ $i \in \{1, 2, 3, 4, 5\}$ satisfies the property that α is a **superkey** of the respective table.

4. The tables which do not have any nontrivial key are (by definition) in BCNF.

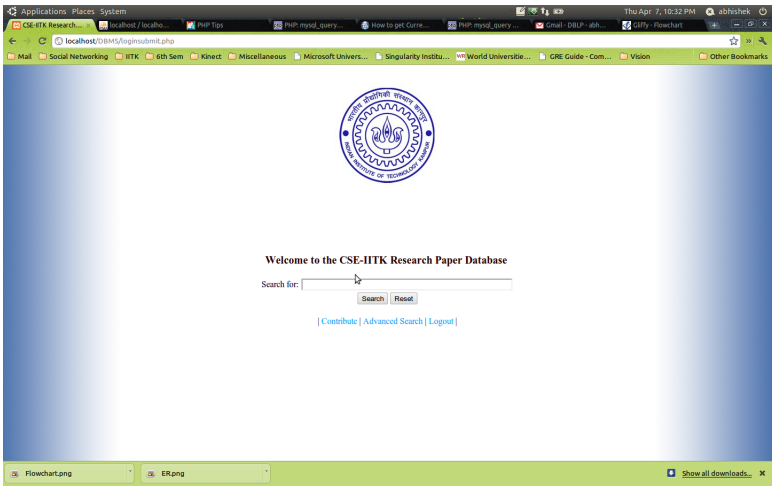
Hence, our design of database is in **BCNF(Boyce-Codd normal form)**i.e. every table is in BCNF.

Snapshots of the Interface

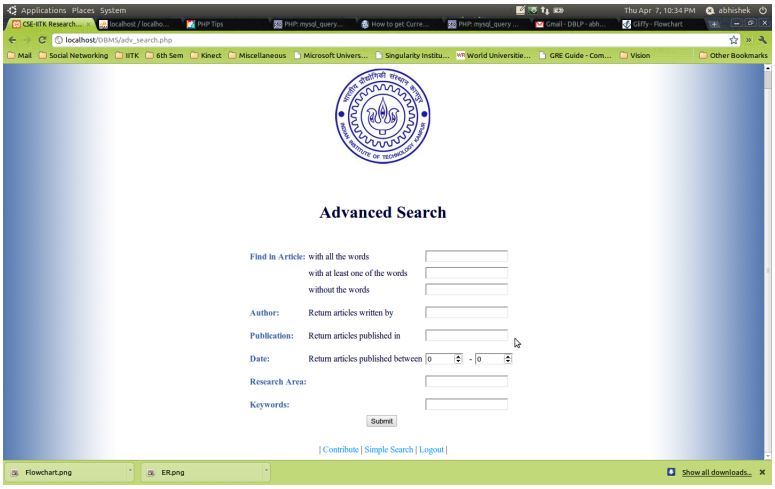
- Login Page



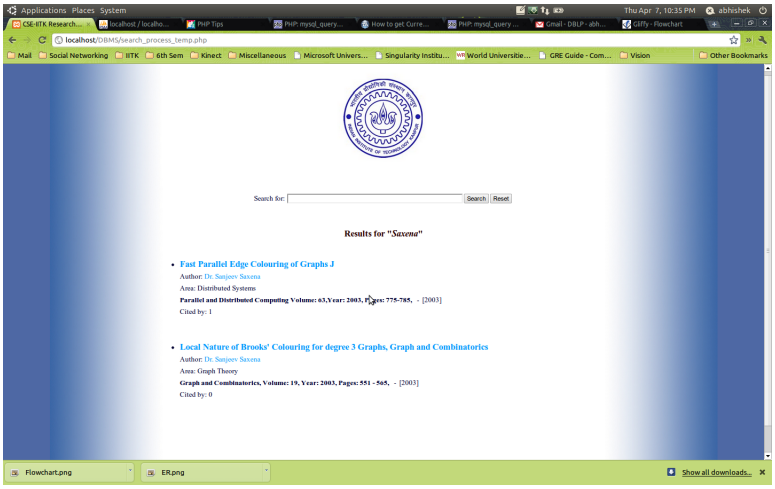
- Simple Search



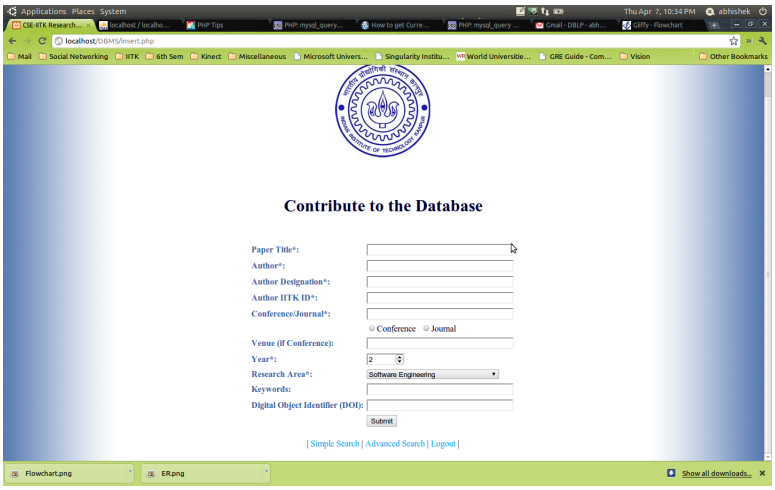
- Advanced Search



- Search Result



- Insert Page



Future Work

Currently we only have access to the BibTex entries of the papers. As the service is internal to IITK, we can obtain the actual papers and have a server to store them. Uploading rights can be given to specific users. This will increase the functionality and ease of use.

References

1. CSE department publications
2. Digital Object Identifiers (DOI)
3. DBLP
4. The Collection of Computer Science Bibliographies