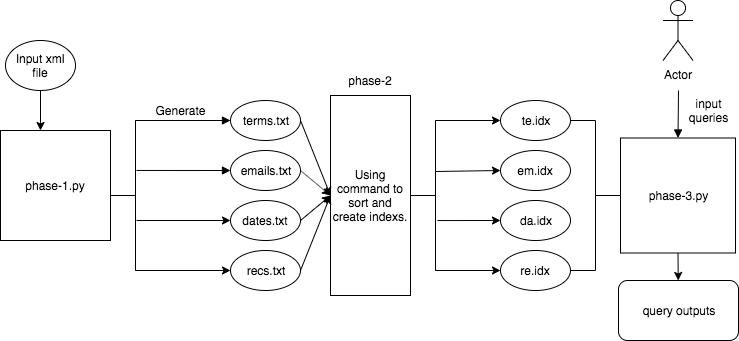
General Overview

The main idea of this program is to use Berkeley DB working with data in physical layer. There are 3 parts of the program. The phase-1.py will output terms.txt, emails.txt, dates.txt and recs.txt based on an input XML file. phase-2.py \_\_\_\_\_\_\_\_\_ and phase-3.py is the program that handles all the input queries.



#####The general overview of the system gives a high level introduction and may include a diagram showing the flow of data between different components; this can be useful for both users and developers of your application.##############

User guides

#############instructions for running your code for phases 1, 2 and 3.#########

For phase 1: run the phase-1.py on terminal or using command “python3 phase-1.py 1k.xml(or put the name of whatever the xml file you want to use)”. This program will generate 4 text files: terms.txt, emails.txt, dates.txt and recs.txt

###current version, should be a makefile or some codes to do those#######

//For phase 2, we sort text files and create index files manually. (1) Sort recs.txt using “sort -nu recs.txt -o recs.txt” and then use perl to break down sorted file “perl break.pl < recs.txt > recs\_formatted.txt”. Then use “db\_load -T -c duplicates=1 -f recs\_formatted.txt -t hash re.idx” to create hash index for recs.txt. (2) Use “sort -u terms.txt -o terms.txt” then “perl break.pl < terms.txt > terms\_formatted.txt” to break and “db\_load -T -c duplicates=1 -f terms\_formatted.txt -t btree te.idx” to create b+tree index for terms.txt (3)Use “sort -u emails.txt -o emails.txt” and “break.pl < emails.txt > emails\_formatted.txt” then “db\_load -T -c duplicates=1 -f emails\_formatted.txt -t btree em.idx” to create index file. (4) Using command: “sort -u dates.txt -o dates.txt” to sort dates.txt, then use perl to break down sorted file: “perl break.pl < dates.txt > dates\_formatted.txt”. After that, we use “db\_load -T -c duplicates=1 -f dates\_formatted.txt -t btree da.idx” to create b+tree index file for dates.txt.//

For phase 3: The default of the output of each query is the row id and subject fields of all matching emails. If user input “output=full”, the format of the outputs will be full records and using “output=brief”, the output will go back to the format of row id and subject fields. For input queries, the format of inputs should strictly follow the formats on eclass “<https://eclass.srv.ualberta.ca/mod/page/view.php?id=3659785>”.

Algorithm description

a description of your algorithm for efficiently evaluating queries, in particular evaluating queries with multiple conditions and wild cards and range searches and an analysis of the efficiency of your algorithm

Testing Strategy

discusses your general strategy for testing, with the scenarios being tested and the coverage of your test cases

For phase 1, we have used 10.xml and 1k.xml to test and compare output text files. However, there is no sample outputs for 10k.xml and 100k.xml, so we can only tested the generate function.

For phase 2, after generate index files, we used db\_dump to visualize index and check it. For instance, for te.idx, use: “db\_dump -p te.idx > te.txt”. We manually check if it is correct. Coverage is 10.xml outputs and some of 1k.xml outputs.

For phase 3, we use output index files of 10.xml and 1k.xml. We checked if the output formats are correct or not, then search input conditions in the text file after db\_dump to check if all matching records covered.

Break-down Strategy

Nasif Hossain (ccid: nhossain) : Implementation and organize codes on phase 1. Fixed errors on emails.txt. Organize commands on phase 2. Break down and reorganize codes on phase 3. Time spent:\_\_\_\_

Yazan AI-Muhtaseb (ccid:\_\_\_\_): Wrote codes for phase 1 for emails, dates, recs. Find sort commands and index commands on phase 2, finished 90% codes of phase 3 by himself. Time spent:\_\_\_\_\_

Pengcheng Yan (ccid: py): Fixed errors on terms.txt, testing codes for phase 1 and report. Time spent:\_\_\_\_\_\_

We use GitHub and Google Drive to keep project on track

Assumptions: Input query formats follow the formats on eclass, input queries are reasonable.

Limitation: