METU, Department Of Computer Engineering

Graduation Project

Proposal Form

(Please read carefully, and follow the instructions to prepare the project proposal form.)

(Instructions to fill in this form are given in italic fonts and in parentheses.)

(To provide an input for a section of the form, delete the instruction and provide your input in place of the deleted instruction. In the final form that you will submit, there shouldn't be any instructions left over, including this section of the form.)

(If you feel that a particular instruction is not relevant to your project proposal, please use a proper explanation for this, rather than ignoring the instruction.)

(The final form should not exceed 4 pages, excluding this page and including the References section. Please use Arial, Normal, 10pt fonts and single line spacing.)

Important Notes

A project could be proposed by (i) a student group, (ii) a company, or (iii) a faculty member of the department by filling in this form and submitting it to 49x-proposal@ceng.metu.edu.tr by e-mail. For a project proposal, there might be a sponsoring company supporting the project and providing some form(s) of resources for the project.

If your proposal might contain a patentable idea or any type of intellectual property, please first make sure to follow appropriate steps (apply for a patent, etc.) before sending your idea to us. Once this form is received from you, the instructor(s) and the department has no responsibility regarding to intellectual properties of your project/idea.

All sources and documentation developed for this course are assumed to be public domain (GPL, CC or similar license) by default. If you need any exception for license and disclosure of project work, please specify this in detail in IP section of the form.

Please note that source codes, documents and issue tracking should be kept in department servers. No restrictions can be requested for limiting faculty and assistants access to student work.

Project Information

Title

Indexing mechanism for RegEx Search

Target

Public [✓]   Restricted [ ]

(If you would like to restrict your project idea to one or more groups, please mark "Restricted" and state the group or groups eligible for the project.)

Proposer Information

Name(s)	***SAP Development Center Turkey,*** *Onur Deniz*
E-Mail(s)	*onur.deniz@sap.com*

IP (Intellectual Property) Information

(Include information about how the project group -and possibly the sponsor- agreed on the intellectual property rights of the end-products –if any.)

Project Description and Background Information

Description

Regular expressions are widely used in a variety of tasks such as text processing, textual data mining, data validation, data scraping and simple parsing [1]. Although there are different implementations of RegEx libraries and applications within different layers such as command line tools (grep[2], sed[3]), scripting languages (Perl[4], Python [5]), common programming languages (java[6], c++[7][8], etc.), few implementations (MySQL[9], PostgreSQL[10], Oracle[11]) exist in data tier(i.e.. database layer). However, to our knowledge none of the implementations above take advantage of an index structure, which results in searching and matching operations to be executed on the fly over all input string sequences.

This project aims to develop a regular expression engine which uses an (k-gram) index structure to accelerate operations over large number of documents. A short research reveals similar studies of using indexes [12][13][14], but none of the databases utilized such kind of specialized index for regex queries.

Similar Products/Projects

Beside all other regular expression implementations on the market, the most similar project is Google Code Search [15] project which is implemented in Go Language. It uses trigram indexes to boost its search operations.

Justification of the proposal

Since regex features of databases on the market are provided without proper index mechanisms, queries should run over all of the documents / rows in the database. And this causes slow response time when the database has many numbers of textual data elements. Output of the proposed project should address this handicap using proper index mechanisms and should fasten query response times for regex queries.

Contributions, Innovation and Originality Aspects of the Project

This project will be different from Google Code Search [15] because of its low level implementation. Code Search was implemented in Go Language, therefore multi-threading issues of the implementation is mostly handled using high level structure of the Go Language. But this project should be implemented in low level languages (C / C++) because the outcome is aimed to be utilized by a database directly.

Technical Aspects of the Project

Study should start with research about implementations and proposals of regex engine in literature, both technical and academic aspects should be covered. Survey should include most common approaches (Nondeterministic Finite Automata and Backtracking) with their example implementations and overall algorithmic designs.

Main (tentative) modules of the products shall be:
Indexing: Deciding on index structure, reading and writing indexes for input streams/strings
Querying: Conversion of regular expression into queries of AND/OR expressions, applying transformations in order to keep the resulting query simple as possible [14], matching resulting query with index
Benchmarking: Extensive benchmarking the product with other regex engines in market [16]

Targeted Output, Targeted User/Domain Profile

End-product will be a desktop application or command line tool. It should also be used as a library in other applications.

It should meet POSIX extended standard, but no need to cover non-regular language features like back-references.

Possible users of the output are developers, architects, text analytics, data scientists etc.

Project Development Environment

C++11, Linux.

Python for testing and benchmark.

External Support

We plan to get some consultancy such as know-how, code-reviews, designing suggestions etc. from SAP Turkey's engineers and our supervisor.

References

[1] https://en.wikipedia.org/wiki/Regular_expression
[2] http://www.gnu.org/software/grep/manual/grep.html
[3] http://www.unix.com/man-page/linux/1/sed/
[4] http://perldoc.perl.org/perlre.html
[5] https://docs.python.org/2/library/re.html
[6] http://docs.oracle.com/javase/7/docs/api/java/util/regex/Pattern.html
[7] http://en.cppreference.com/w/cpp/regex
[8] http://www.boost.org/doc/libs/1_59_0/libs/regex/doc/html/index.html
[9] https://dev.mysql.com/doc/refman/5.1/en/regexp.html
[10] http://www.postgresql.org/docs/9.3/static/functions-matching.html
[11] https://docs.oracle.com/cd/B12037_01/appdev.101/b10795/adfns_re.htm
[12] http://oak.cs.ucla.edu/~cho/papers/cho-regex.pdf
[13] https://wiki.postgresql.org/images/6/6c/Index_support_for_regular_expression_search.pdf
[14] https://swtch.com/~rsc/regexp/regexp4.html
[15] https://github.com/google/codesearch
[16] http://lh3lh3.users.sourceforge.net/reb.shtml