Introduction

In this short report we are going to understand how Russ Cox implemented the google code search package [1] that actually went online and was serving most of google code until it was shut down in 2011. Apart from full text search he also implemented a regular expression search option on top, which was optimized by the FTS engine. In the following sections we will investigate how it worked.

Indexed word search

Beginning with a basic inverted index and the following three simple documents

(1) Google Code Search
(2) Google Code Project Hosting
(3) Google Web Search

The inverted index for these three documents looks like:

Code:	1, 2
Google:	1, 2, 3
Hosting:	2
Project:	2
Search:	1, 3
Web:	3

To find all the documents that contain both Code and Search, you load the index entry for Code 1, 2 and intersect it with the list for Search 1, 3, producing the list 1. To find documents that contain Code or Search (or both), you union the lists instead of intersecting them. Since the lists are sorted, these operations run in linear time.

To support phrases, full-text search implementations usually record each occurrence of a word in the posting list, along with its position:

Code:	(1, 2), (2, 2)
Google:	(1, 1), (2, 1), (3, 1)
Hosting:	(2, 4)
Project:	(2, 3)
Search:	(1, 3), (3, 4)
Web:	(3, 2)

To find the phrase “Code Search”, an implementation first loads the list for Code and then scans the list for Search to find entries that are one word past entries in the Code list. The (1, 2) entry in the Code list and the (1, 3) entry in the Search list are from the same document (1) and have consecutive word numbers (2 and 3), so document 1 contains the phrase “Code Search”.

Trigrams

The process of seperating a stream of elements like a string to all possible n-sized sequential elements is to give us the n-grams of the sentence. A trigram is just the n-gram with n=3. Under this construction the group of all trigrams for the sentence "the quick red" has the following character-level trigrams (where an underscore "_" marks a space):

`the,he_,e_q,_qu,qui,uic,ick,ck_,k_r,_re,red`

By observing that when given a regular expression such as /Google.*Search/, he could build a query of ANDs and ORs that gives the trigrams that must be present in any text matching the regular expression. In this case, the query is

Goo AND oog AND ogl AND gle AND Sea AND ear AND arc AND rch

Then he could run this query against the trigram index to identify a set of candidate documents and then run the full regular expression search against only those documents.

Summary

By using trigrams to constrain the number of documents on which the code-search is looking for a regular expression match, Russ Cox was able to push out a working version of codesearch that served millions of lines of code from 2006 till its retirement around 2011. The essence of his algorithm was to extract the trigrams from the regexp and first look up the inverted index for the correct documents, and only then apply the full regexp match on those documents. Later he even published a go open source implemetation that explored these ideas and could run locally on a users' computer [2].

Bibliography

[1] Regular Expression Matching with a Trigram Index , Russ Cox . `https://swtch.com/~rsc/regexp/regexp4.html`

[2] Google codesearch `https://github.com/google/codesearch`