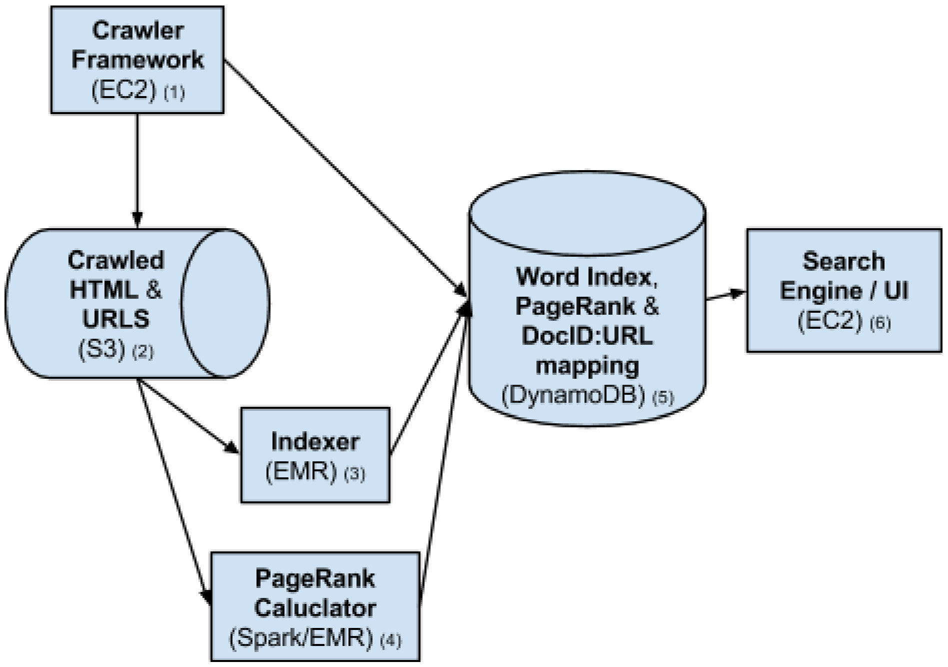
CIS555 Project Plan

Introduction

We are creating a search engine that will be hosted on Amazon's AWS ecosystem. The search engine will return relevant results as determined by both our implementation of the PageRank algorithm and a word index generated from over 100,000 web documents.

Architecture and Approach



Crawler Framework (EC2) (1)
Crawled HTML & URLS (S3) (2)
Indexer (EMR) (3)
PageRank Caluclator (Spark/EMR) (4)
Word Index, PageRank & DocID:URL mapping (DynamoDB) (5)
Search Engine / UI (EC2) (6)

We will run a number of **crawlers** on EC2 (1), which will store crawled HTML, XML, text or PDF documents, as well as a document containing URLs_from and URLs_to in S3 (2). The crawler system will also store a DocID:URL mapping table in DynamoDB (5). Each crawler will be responsible for a particular subset of domains (based on each domain's hash value), and will assign a unique DocID to each URL encountered (regardless of whether the document is crawled).

The **indexer** (3) will run an EMR job on the crawled documents on S3 (2), and generate an index of stemmed words (using a Porter stemmer). Each word occurrence in each document will be assigned a Hit object, which will be in the form of an integer that encodes the nature of the word's occurrence (in the title, in an anchor tag etc), the position of the word in the text, and whether the word is capitalized. The indexer will also calculate the normalized term-frequency for each word in each document, and well as the inverse-document-frequency for the word. The resulting inverted index will be stored on DynamoDB (5).

The **PageRank calculator** (4) will retrieve a URLs_from: URL_to from mapping from S3 (2). It will then run an iterative MapReduce job (either using Spark or EMR's raw Hadoop framework), finding the principal eigenvectors of the (URL_from, URL_to) matrix in order to generate a ranking of pages. The resulting PageRank score will be stored on DynamoDB (5).

Given a user query, the **search engine** (6) will use the inverted index stored on DynamoDB (5) to gather a pool of all relevant DocIDs, i.e. documents (URLs) that contain at least one of the words in the query. It will use the each word's term-frequency for the relevant DocID, as well as the word's inverse-document-frequency value (also stored on DynamoDB) to calculate the cosine similarity of each DocID with respect to the query .

In addition to the cosine similarity, the search engine will calculate its final ranking based on each candidate DocID's PageRank (stored on DynamoDB), as well as a list of Hit objects for each word-DocID pair. The weights assigned to each factor to determine the optimal results will be obtained through experimentation.

Roles

Crawler & EC2 / EBS research: Kevin Lee
Indexer & EMR research: Yunchen Wei
PageRank & EMR / Spark research: Rohit Gupta
Search Engine & DynamoDB research: Joop Hong

Milestones

April 18th:
- Complete single node crawler
- Complete indexer for different document types (HTML, XML, text, PDF)

April 25th:
- Store crawled documents / URL files stored on S3 and DocID:URL mapping on DynamoDB (from a single node Crawler)
- Set up EMR framework for indexer
- Set up EMR framework for PageRank calculator
- Set up basic UI and servlets for the search engine

April 28th:
- Complete multiple node crawler
- Build inverted index on DynamoDB
- Generate PageRank for documents using iterative MapReduce process.
- Run preliminary analysis on PageRank results
- Experiment with factor weights for the search engine
- Generate improved version of UI

May 2nd:
- Integrate all components, final tweaking of search engine results