Commentary on Application of Hidden Markov Model in Google Page Ranking

Prerna Rai
Sikkim Manipal University
Department of CSE, SMIT, Sikkim

Moirangthem Goldie Meitei
Sikkim Manipal University
Department of CSE, SMIT, Sikkim

Ferdousi Khatun
Sikkim Manipal University
Department of CSE, SMIT, Sikkim

Abhijit Choudhury
Sikkim Manipal University
Department of CSE, SMIT, Sikkim

Udit Kr. Chakraborty
Sikkim Manipal University
Department of CSE, SMIT, Sikkim

ABSTRACT
In this document, the Google Page ranking and the algorithms behind this technique have been put up. Google, in its efforts to crawl and index the Web efficiently and produce much more satisfying search results than other existing systems, has been continuously working on optimizing its search results. This optimization is done by a mechanism called PageRank. Page ranking for Google has been implemented using Markov chain and we present here the use of Hidden Markov Model (HMM) in Google page ranking and its implications. These algorithms are used to search and rank websites in the Google search engine. PageRank is a way of measuring the importance of website pages. Markov chain model and Hidden Markov Model are mathematical system models. They describe transitions from one state to another in a state space. The Markov model is based on the probability the user will select the page and based on the number of incoming and outgoing links, ranks for the pages are determined. HMM also finds its application within Mapper/Reducer. These algorithms are link analysis algorithms. The current paper presents an analytical study on the application of HMM in Google Page Ranking.

General Terms
Stochastic model, link analysis algorithm

Keywords
Damping factor, Links, Markov model, PageRank, Probability, Web-graph, Stochastic

1. INTRODUCTION
Google is the largest and most widely used search engine on the web. Each day it provides services to millions of queries that it receives from Internet users. Google organizes huge amount of information and makes it universally accessible. The main aim of this search engine is to return the page being queried in order of their preference. The first ten pages returned by the search engine are the most relevant pages and are of most importance to the user. The technique used by Google to rank web pages according to their importance is called PageRank, developed by Larry Page and Sergey Brin at Stanford University. PageRank works by counting the number and quality of links to a particular page in order to find out a rough estimate of the importance of website. The Google search engine works with an assumption that websites that receives more links from other websites have higher importance and are thus given higher rank.

Underlying this apparently simple idea are a bunch of related techniques that are used. They are Markov model, Markov chain model and Hidden Markov model. Markov models are widely used to model sequential processes, and have achieved many practical successes in areas like web log mining, speech recognition, natural language processing, robotics, etc. Using these techniques Google computes the rank of web pages and then determines the order of the pages in which it should be listed in search results provided by the search engine. These models, Markov model or Markov chain model and Hidden Markov model, are stochastic finite state machines. Stochastic here means random or based on theory of probability. A stochastic process is a process whose behaviour is nondeterministic and the system's subsequent state is determined both by the process's predictable actions and by a random element.

2. RELATED APPROACH
Google search engine responds to the user's query in accordance to their importance using various techniques. And the techniques that have been discussed here are:

1. Google Page Ranking.
2. Markov Model & Markov Chain Model
3. Hidden Markov Model.

Google PageRank algorithm as per [8] assumes a probability distribution between 0 and 1.It works by assigning score to number of web pages on World Wide Web and hence when a user requests for any page from the search engine, it returns the pages based on their score. And hence we can find that the user's search is returned with the importance of the pages and most often we find that the first top ten search results provided by Google may be enough for fulfilling the user's request.

The other technique used by Google to rank pages is by using Markov Model. It uses the concept of Markov chain. It is used to predict the behaviour of the system that moves from one state to another. The state space of the model depends on the current state and not on the sequence of events that preceded it. This model is a stochastic finite state machine.

Hidden Markov Model is a model that has three distinct stages of evaluation, decoding and training for the two distinct states. They are hidden and observed states. To determine the hidden states from the observed state it uses the Viterbi algorithm, based on the Forward Algorithm. It traces the most likely hidden states while reproducing the output sequence.

3. GOOGLE PAGERANK ALGORITHM

Google uses the basic PageRank algorithm to perform link analysis [6]. Link analysis algorithms for web search engines determine the relevance of web pages. Among many link analysis algorithms PageRank is one of the mechanisms used by Google search engine [16].

PageRank is a query and content independent algorithm [16]. Content independent means the PageRank algorithm does not include the contents of web page for ranking .It uses the link structure of the web and assigns page rank score to each web pages on WWW using mathematical model. When a query for a page is made by the Internet user, Google search algorithm compiles the page rank scores using page rank algorithm and hence returns the pages depending upon the score of each page being queried. The basic PageRank algorithm outputs a probability distribution. The transition matrix that PageRank algorithm considers the web as a directed graph, where nodes represent the pages and edges represent the hyperlinks in the page.

PageRank algorithm for Google was developed by Page and Brin, and today this algorithm has been the heart and soul of Google search engine. The pages are ranked offline and are content independent.

The name PageRank states that the pages in the web are assigned a score which is assigned as a numerical value and for any suitable page P, a PageRank is denoted by PR(P).The rank of page is based on mathematical algorithm based on web-graph, created by World Wide Web pages as nodes and hyperlinks as edges. The algorithm has its basis on the probability distribution. The score of the page indicates the importance of the page. More the hyperlinks from the other important pages higher will be the score and hence higher the score of the page higher will be the rank of a page.

The PageRank computations require several “iterations” (until it converges) and are determined probabilistically. A probability is expressed as a numeric value between 0 and 1. A 0.5 probability is commonly expressed as a "50% chance" of something happening. Hence, a PageRank of 0.5 means there is a 50% chance that a person clicking on a random link will be directed to the document with the 0.5 PageRank.

3.1 Simplified form of Google Page Rank Algorithm

In a web server there are millions of pages and out of all these many pages the search engine needs to return the pages to the user based on the user’s need. So to do so how does Google respond to the user’s query that meets the users need? This is done using page Ranking, developed by Page and Brin. During their works they came up with many algorithms to rank the page according to score and this is among one of their findings.

When a page score needs to be determined in general we cannot consider the surfer to select the pages sequentially. There may arise a situation where the surfer may click the pages randomly.

Though most of the time, a surfer will follow links from a page, from a page i the surfer will follow the outgoing links and move on to one of the neighbours of i. But this may not happen always, a smaller, but positive percentage of the time, the surfer will dump the current page and choose arbitrarily a different page from the web and "teleport" there.

So to overcome such situations Page and Brin introduced a factor called as the damping factor d, that reflects the probability that the surfer quits the current page and "teleports" to a new one. Since he/she can teleport to any web page, each page has 1/n probability to be chosen.

In general, to compute page rank PR of page P the formula given by Page and Brin is

$$PR(P) = \frac{(1-d)}{N} + d\left[PR\left(\frac{P_i}{O_i}\right)\right] \quad (1)$$

Here d is a damping factor such that 0<=d<=1 and Oi is the number of outgoing links of page i. The damping factor d is given as values 0.85.

In order to explain simplified Google PageRank algorithm, the following assumptions are made [8]:

a. Three web pages: A, B, C are considered

b. Links from a page to itself, or multiple outbound links from one single page to another single page, are ignored.

c. PageRank is initialized to the same value for all pages.

Consider an example of a web pages having following links:

• page A has a link to pages C and B

• page C has a link to page A, and

• page B has links to all A and C pages.

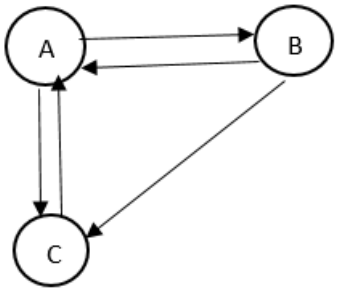

Figure 1 PageRank example

In this scenario during the first iteration the PageRank of A, B, C will be calculated as follows:

PR(A)=0.15/3+.85 [PR(B)/2+PR(C)/1]=0.432

PR(B)=0.15/3+.85[PR(A)/2]=0.233

PR(C)=0.15/3+0.85[PR(A)/2+PR(B)/2]=0.3

This computation continues until PageRank gets converged or there do not exist further changes and until the sum value is not equal to 1.

Based on the score it is found that A has highest score and B has the least. This is due to the reason that A has two outgoing links and 2 incoming links. B has one incoming and 2 outgoing. C has two incoming and one outgoing. Therefore more the number of incoming links higher will be the score. And in this scenario Google would be selecting page A and responding to the user.

Therefore it is seen that one page's PageRank is calculated by the PageRank of other pages. Google is always recalculating the Page Ranks. If all pages are given a PageRank of any

number (except 0) and constantly recalculate everything, all Page Ranks will change and tend to stabilize at some point. It is at this point where the PageRank is used by the search engine. This algorithm is the original and simplified algorithm used by Google search engine. Google has also been using other algorithms to rank pages and determine pages for the selection by the user. Below, the other algorithms, which Google uses, have been discussed. They are Markov Chain Model and Hidden Markov Model.

4. MARKOV MODEL

Markov model was first introduced by Russian Mathematician Andrey Markov. The Markov model is a stochastic model to predict the behaviour of a system that makes a transition from one state to another where the future state is depends only on the present state. Markov model has wide range of applications in various fields of science and engineering, but the most recent application of this model is on the Google search Engine. Google uses the concept of page ranking as discussed above and Markov model also finds its major role in page ranking of Google search engine.

As in [2] Markov chain is a random process where all information about the future is contained in the present state. To determine the future the past examination is not necessary. The model can be represented by FSM [7] with their states and transition that transit from one state to another. This model is a Non Deterministic Finite State Machine. The state at each step can be predicted by a probability distribution associated with the current state. It is a mathematical system model that describes transitions from one state to another on a state space. The state space of the model depends on the current state and not on the sequence of events that preceded it. This is also known as the Markov property.

Markov chain is the sequence of random variables. Markov Chains models the adjacent period that is used for describing a system that follow a chain of linked events, where what happens next the sequential processes. It assumes a discrete random variable q with states. State of q changes randomly in discrete time steps. Transition probability depends only on the k previous states. In the Markov chain Random transition Probability depends only on the prior states time .In fig 2 A,B,C,D are the state of a system and each has some probability .If the probability is 2/3 in state A then it makes a transition to state B.

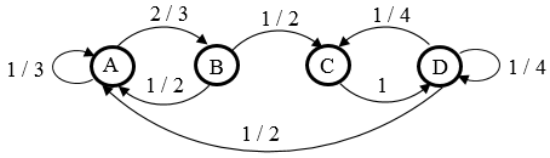

Figure 2 Markov Chain

A Markov Model can have Nth order [2].

- 0th order models depend on no prior state.
- 1st order models depend on one previous state.
    - If k states, need to specify k2 transition probabilities
    - k x k
- nth order models depend on n previous states.
    - If k states, need to specify k(n+1) transition probabilities
    - (k x k x k … x k)/n

Markov models have the following properties:

- Behaviour at time t depends only on its state at time t-1.
- Sequence of outputs produced by a Markov model is called a Markov chain
- Process of Markov chain model in order to perform page Ranking:

In a system at any given time t = 1, 2, 3…n occupies one of a finite number of states. At each time t the system moves from state v to u with probability P_{uv} that does not depend on t.

P_{uv} is called as transition probability and this is what determines the next state of the object considering only the current state.

4.1 Transition Matrix

Transition Matrix T is n x n matrix. n represents the number of states. The matrix is formed from transition probability of Markov process. Each entry in the transition matrix t_{uv} is equal to the probability of moving from state v to u at time t. Therefore $0 <= t_{uv} <= 1$ must be true for all u and v.

For example, consider the following:

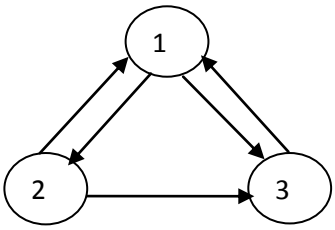

Figure 3 A connection of three pages

There are three states or pages {1,2,3}.

The transition probability of this graph is

$$t_{uv} = \begin{pmatrix} 0 & .5 & 1 \\ .5 & 0 & 0 \\ .5 & .5 & 0 \end{pmatrix}$$

This matrix determines the probability of making a transition from one state to another. There are three state 1,2 and 3.1 reaches 2 and 3 with probability of 0.5 each. From state 2 it reaches 1 and 3 with 0.5 probability and state 3 reaches 1 and 2 with 0.5 each.

The transition Matrix has the following properties:

1. The matrix must be non-negative and should be square matrix i.e. the no of rows should be equal to the no of columns. Each row and column represents state.
2. The entries in the matrix represent probability i.e. it should be within 0 and 1.
3. The sum of entries in a row is the sum of the transition probabilities from one state to another state, therefore the sum of row value should be equal to 1.This kind of matrix is called stochastic matrix.

As per [2] formally, a Markov model is a triple M = (K, π, A)

Where

- K is a finite set of states.(in our example we have 1,2,3 states)
- π is a vector of initial probabilities of each of the states. Here in our example vector is a matrix of 3rows with

$$\pi = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix}$$

- A [u, v] = Pr(state v at time t | state u at t - 1) the probability that, if M is in u, it will go to v next.

One of the major internet applications of Markov model is PageRank of a webpage as used by Google. It is defined by a Markov chain. It is the probability to be at page i in the stationary distribution on a Markov model on all webpages.

The states determined above can be considered as web pages. Imagine random surfer surfing the web, selecting one page and going to other randomly by choosing an outgoing link from a page. This can sometimes lead to page from where there are no outgoing links. So a certain fraction of time the surfer chooses a random page from the web. This theoretical random walk is known as Markov chain or Markov process. This limiting probability that an infinitely dedicated random surfer visits any particular page is its page Rank [16].

To deal with random surfers, Markov chain is being used to rank a page by Google search Engine. The number of links to and from the page helps to find out the importance of any page and hence their PageRank. The more incoming links a page has, the more important the page is. Back links from more important pages carries more weight than back links from less important pages. If good page links to several other pages then the weight will be distributed equally to all those pages [16].

4.2 Generalized page rank using Markov chain

Let C = {c_{ij}} denote the adjacency matrix, a n × n matrix with c_{ij} = 1 if there is an hyperlink from page i to page j and c_{ij} = 0 otherwise.

The outgoing degree of page i, meaning the number of pages that can be reached from page i, will be the row sums, denoted as si (based on the property of Transition matrix stated above):

$$S_i = \sum_{j=1}^{n} C_{ij} \qquad (2)$$

Now normalizing the adjacency matrix C by its row sums, we get $\bar{W}$ = {wij}, defined by

$$W_{ij} = \begin{cases} C_{ij}/S_i, S_i \geq 1 \\ 1/n, \ S_i = 0 \end{cases} \qquad (3)$$

This models the situation when the internet user is at page i. If there are S_i outgoing links on page i, the user will pick one, with equal probability, as its next page to visit. If there is no outgoing link on page i, the user will pick a random page. This is a simplified model of the user behaviour.

In a transition matrix if the sum of any row is equal to 0 than the node is considered to be a dangling nodes or hanging nodes. These are node having no outgoing links. Dangling nodes cannot present in the web if it is to be presented using a Markov model. So to overcome this problem dangling nodes need to be eliminated. Langville et.al.[16] proposed a method to handle dangling nodes by replacing all the row value with 1/n, where n is the number of states or nodes. This makes the transition matrix stochastic matrix.

The Markov chain can also be elaborated by introducing an additional tuning parameter or damping factor γ/d, and probability should be between 0 and 1.

Now, the transition probability matrix of the Markov chain used in PageRank is given as follows:

$$P = \gamma \bar{W} + (1-\gamma)\frac{1}{n} E \qquad (4)$$

where E is the n × n matrix with all entries being 1. This Markov chain describes the behaviour of an internet user who, with probability γ follow an outgoing link on the current web page with equal probabilities or, if the page has no outgoing links, jumps to another page randomly. Also, with probability 1−γ, the user jumps to a page randomly with equal probability. On proper choice of γ, this Markov chain is finite, irreducible also aperiodic, and there exists a unique stationary distribution π. This stationary distribution is used to rank all the pages in W: the page i with the largest π_i will be ranked first, the second largest be ranked second, and so on. In the computation of PageRank, γ is usually set to 0.85.

The process continues or iterates till the matrix converges and this convergence is possible only if the entries to the transition matrix satisfy 0 <= t_{uv} <= 1.

5. HIDDEN MARKOV MODEL

A Markov process with hidden states is called Hidden Markov Model. It is a nondeterministic finite state machine. In a simpler Markov models (like a Markov chain), the state is visible to the user, and therefore the state transition probabilities are the only parameters whereas in Hidden Markov model there are two basic parameters, observed and hidden. The hidden state is not directly visible, but the output, dependent on the state, is visible. Each state has a probability distribution over the possible output tokens. Therefore the sequence of tokens generated by an HMM gives some information about the sequence of states. The effect of the hidden states may be observed in HMM.

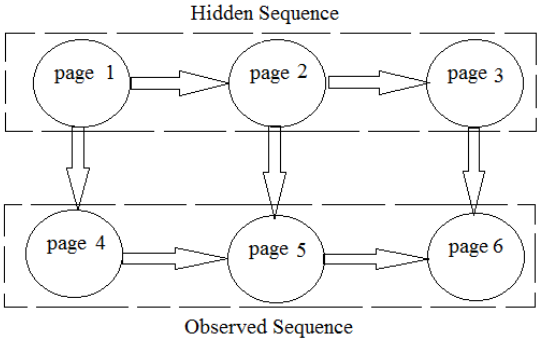

Figure 4 Hidden Markov Model

This model provides three kinds of probabilistic information [21].

Probability at the start of the machine

If p and q are the states than the transition from state p to q is labelled with probability that it will go to the next state. If no

transition than the probability is 0. Each output state from q is labelled with the probability of reaching final state.

Formally according to the study in [2] an HMM M is a quintuple (K, O, π, A, B), where:

- K is a finite set of states,
- L O is the output alphabet,
- π is a vector of initial probabilities of the states,
- A is a matrix of transition probabilities:
- A[p, q] = Pr(state q at time t | state pat time t – 1),
- B the confusion matrix of output probabilities.
- B[q, o] = Pr(output o | state q).

HMM is a tool to find underlying processes to a given sequence. It is used to model situations like the transition of states with random variable p based on the current state of q, where q is the hidden state. Hidden state affects the observed variable p.

There are three main problems for a HMM as stated by Rabiner [18]:

- Evaluation: Given an observation sequence O and a set of HMMs that describe a collection of possible underline models, choose the HMM that is most likely to have generated O.
- Decoding: Given an observation sequence O and HMM M, discover a path through M i.e. most likely to have produced O.
- Training: How we adjust π, A, B so that the resulting HMM has the property that, out of all the HMMs whose states set is equal to K, M is the one most likely to have produced the outputs that constitutes the training set.

The most likely hidden path can be computed efficiently using a dynamic programming approach called the [21] Viterbi algorithm which solves the decoding problem.

The Evaluation problem is solved by the [21] forward algorithm which computes the probability that the chosen HMM M could have output O along any path.

The training problem [21] can be solved by following an iterative procedure such as the Baum-Welch method.

5.1 Viterbi Algorithm as proposed by Rabiner

The Viterbi algorithm consists of the following steps [18]:

1. Initialization
2. Recursion
3. Termination
4. Path Backtracking

The Viterbi algorithm is a dynamic programming algorithm for finding the most likely sequence of hidden states – called the Viterbi path – that results in a sequence of observed events, especially in the context of Markov information sources and hidden Markov models. The Viterbi algorithm selects the highest score associated with any path to q. It is commonly used in speech recognition, speech synthesis etc.

5.2 Forward Algorithm

The forward algorithm [21], in Hidden Markov model, is used to calculate a hidden state which is the probability of a state at a certain time, given the observed state. The forward algorithm is closely related to, but distinct from, the Viterbi algorithm. Forward algorithm solves the evaluation problems of HMM. It is quite similar to the Viterbi algorithm except that, instead of finding the single best path through an HMM M, it computes the probability that M could have output O along any path. The forward algorithm computes the sums of all the scores. The forward algorithm does not need to find a particular path, else it will only compute the probability and returns it therefore unlike Viterbi it will not have to bother maintaining the back pointer array.

5.3 Baum-Welch

Baum Welch [21] is an iterative procedure. It aims to find unknown parameters of HMM. It makes use of the forward-backward algorithm. The main goal of this algorithm is to tune π, A, B so that the resulting HMM M has the property that, out of all the HMMs whose state set is equal to K, M is the one most likely to have produced the outputs that constitutes the training set.

Baum-Welch algorithm [21] employs a technique called Expectation Maximization to find the maximum likelihood. Using this technique it finds the parameters of a HMM given a set of observed feature vector. EM begins with some initial set of values π, A, B. Then it runs the forward algorithm along with a related backward algorithm on the training data. The result of this state is a set of probabilities that describes the likelihood that the existing machine with the current values of π, A, B, would have output the training sets using those probabilities Baum Welch updates π, A, B to increase those probabilities. This process continues until no changes to the parameter value can be made.

HMMs [13] can be applied whenever an underlying process generates sequential data: It can be used for Speech Recognition, Handwritten Letter Recognition, Part-of-speech tagging, Genome Analysis, Customer Behaviour Analysis, Context aware Search etc. Typically, HMMs can also be used within Mapper Reducer.

5.4 Use of HMM in Google Page Rank

Hidden Markov Models can be used to predict a web user's probability of performing a task. One such application of this is in e-commerce where users' browsing data can be collected to infer certain behavioural characteristics of the users. For example, HMM can be applied to determine a user's behaviour in making an online purchase [19]. Detailed study on HMMs have also been made to associate behaviour patterns exhibited by online shoppers while making a purchase [20]. These examples can easily be applied to a web surfing scenario, where HMM can be used to predict a web surfer's behaviour, i.e. which links he is likely to navigate.

The main focus of our study here is the use of HMM in page ranking. From the study of HMM above it has been found that unlike Markov chain the states of the system are not correctly observable. Instead the model has a separate set of output symbols which are emitted with a specified probability whenever the system enters one of its hidden states.

As mentioned above, HMM uses Viterbi algorithm, Forward algorithm and Baum Welch algorithm to solve the problem of decoding, evaluation and training problem. This solution can be used by Google to rank page.

From the study, we find that the use of forward algorithm can be applied in determining the score of the page by computing the probability of the number of web pages that could have determined the output O along any path.

Forward [21] (M: Markov Model, O: output sequence)

1. For t=0, for each state q, set forward score [q,t] to π[q].
2. Find the total probability associated with each state.

For t=1 to |O| do:

2.1. for each state q in K do:

2.1.1. consider each state p in K preceding q:

Candidatescore[p] = forwardscore[p,t-1] * A[p,q] * B[p,O]

2.1.2. sum scores:

Forwardscore[q,t]=∑candidatescore[p].

3. Find the total probability of going through M along any path, giving state of M and emitting O.

Totalprob=∑forwardscore[q,|O|]

Return Totalprob.

This algorithm can be executed in all the contending HMMs and final highest score can be returned.

Hence we can infer from the above algorithm that Google can use HMM for further ranking a page even for those states which are hidden and are not able to be determined by Markov chain. We can consider this method as an extension of Markov chain model and being used extensively by Google to find the score of each page and return the page with the highest score and respond to the search query the user provides to the search engine.

There are other applications also that use Hidden Markov Models and are often compact. The trained HMMs can be efficiently used within Mappers/Reducers. It provides an approach to parallelizing learning.

On account of this, [17] Google is using the approach of Map Reduce. It is a search engine which processes large amounts of raw data, such as crawled documents, web request logs, structure of web documents, summaries of the number of pages crawled per host, the set of most frequent queries in a given day, etc. For the large computation the input data is a huge and distributed across hundreds or thousands of machines. The issues of how to parallelize the computation in an abstraction is one major focus of Google, and to perform this it uses the map and reduce function.

Mapper takes the huge data record, splits them into nodes and using key/value pair as input, it computes a set of intermediate key/value pairs, and then applying a reduce operation to all the values that shared the same key, in order to combine the derived data appropriately.

The use of a functional model with user specified map and reduce operations allows the search engine to parallelize large computations easily. To carry out this function Hidden Markov Model can be used inherently within the Mapper/Reducer function of Google.

6. CONCLUSION

The concept of "PAGE RANK" determines the importance of webpages. Every web page will be given a rank based on their link structure. It is a probability distribution which is used to represent the likelihood that a person randomly clicking on links will arrive at any particular page.

The study has found that the most widely used search engine i.e. Google, uses Page Rank algorithm and Markov model for determining the score of the page in web server. When the user queries for any page Google returns the page having highest score. This score for each page is determined using PageRank and Markov model. It works on the condition that the users are a random surfer. The other model, HMM also finds its application in many aspects. HMM, an extension to Markov Model and has a separate set of output symbols which are emitted with specified probabilities, whenever the system enters hidden states. This model uses forward algorithm to solve evaluation problem and help in finding the score of each page in the web server. It then returns the page to the user with the highest score. From the study it can also be determined that the search engine have implemented the HMM (Hidden Markov model) for map reduce function or for distributed sorting of data. Mapper Reducer functions are the abstract way of handling a huge about of data that is being collected in the web and for this HMM can be incorporated with Mapper and Reducer to perform parallelization.

7. ACKNOWLEDGMENTS

The authors would like to thank Department of Computer Science and Engineering, Sikkim Manipal Institute of Technology for providing useful guidance to accomplish this survey.

8. REFERENCES

[1] Paul ,B.1990 Speech Recognition Using Hidden Markov Models

[2] Finite State Machines .2000 . Available on http://inst.eecs.berkeley.edu/~cs61c/sp08/labs/10/PH-B10.pdf .

[3] Gales, M. and Young, S.The Application of Hidden Markov Models in Speech Recognition in Signal Processing.

[4] Hidden Markov Models and other Finite State Automata for Sequence Processing, in Institute for Perceptual Artificial Intelligence (IDIAP).2002.

[5] Blei M. D,.Hidden Markov models .2012.

[6] Kumar, G. Duhan, N. and Sharma A. K. 2011.Page Ranking Based on Number of Visits of Links of Web Page at IEEE, International Conference on Computer & Communication Technology (ICCCT).

[7] Li,j.2012 .Markov Chain Interpretation of Google Page Rank.

[8] PageRank, Available:http://en.wikipedia.org/wiki/PageRank.

[9] A Look at Markov Chains and their Use in Google at Iowa State University MSM Creative Component .

[10] The Performance of Page Rank Algorithm under Degree Preserving Perturbations University of Sydney, Australia.

[11] Wu,J.,Aberer and K.Using a Layered Markov Model for Distributed Web Ranking Computation .

[12] Siddiqi.,M.,S.,Byron, S. Gordon and G. J., Reduced-Rank Hidden Markov Models.

[13] Heimel ,M., An Introduction to Hidden Markov Models.

[14] Tibshirani, R. PageRank Data Mining.

[15] An Introduction to Hidden Markov Model .2012 Available on http isabel-drost.de/hadoop/slides/HMM.pdf

[16] Kumar,R., GohKwangLeng,, A.and Singh. A.K Application of Markov Chain in the page rank algorithm Available https://researchgate.net

[17] Dean, J., Ghemawat and S., MapReduce: Simplified Data Processing on Large Clusters .

[18] Rabiner L.R, Juang and B. H. An Introduction to Hidden Markov Models.

[19] Lin, C.J, Wu. F and Chiu I.H.Using Hidden Markov Model to Predict the Surfing User's Intention of Cyber Purchase on the Web.

[20] Montgomery Alan L., Li,S., Srinivasan K, and Liechty John C. 2002. Modeling Online Browsing and Path Analysis Using Clickstream Data .

[21] Stochastic Finite Automata: Markov Models and HMMs.