

Internet Technology
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
Lecture No #39
Search Engines and Web Crawler :: Part 2

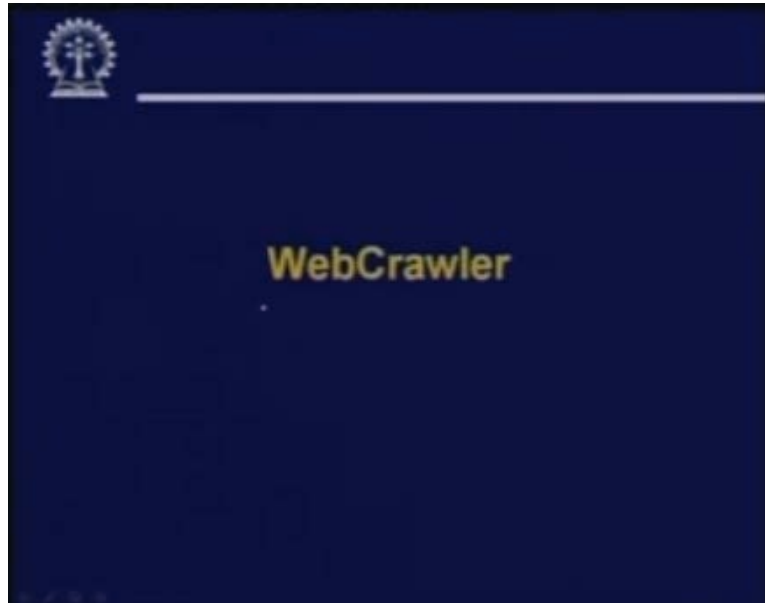
So today we shall continue our discussion on the search engines and web crawlers.

(Refer Slide Time: 01:02)



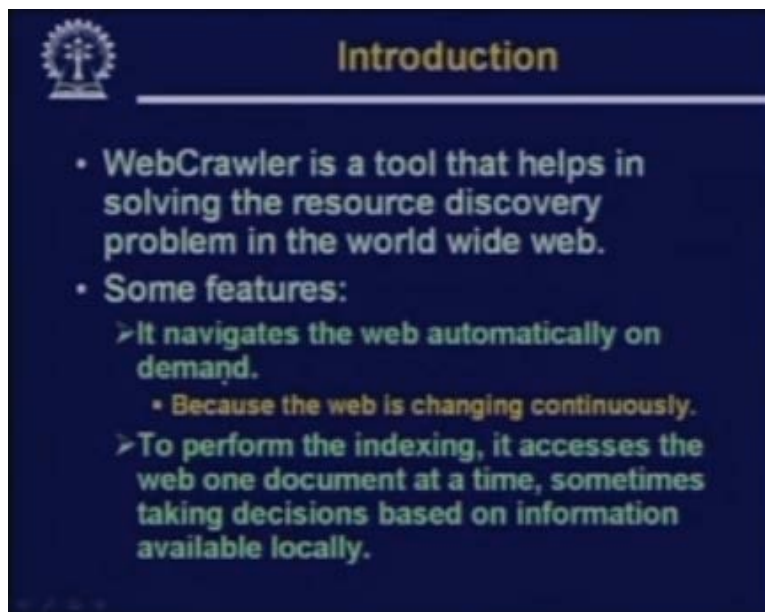
Now if you recall our last day discussion, now last lecture we basically talked about some of the underlying principles which are followed by the search engines. Like how they decide on the relevancy of a page? How they extract keywords from a given document? How they judge the goodness of a document and so on. So, today as I said we shall be looking at the web crawler search engine. How it works and this will possibly give you a good idea about the design or the architecture of a search engine based on that you will be able to design your own search engine if you want to. So we shall be talking about web crawler. So this web crawler, as I said web crawler was a crawler based search engine which was developed in the nineties and it gained quite a bit of popularity. Because at that time crawler based search engines were not that readily available and many of the other existing commercial search engines they used the web crawler services to integrate the search results in to the web pages.

(Refer Slide Time: 02:34)



So web crawler is the topic of our discussion today.

(Refer Slide Time: 03:37)

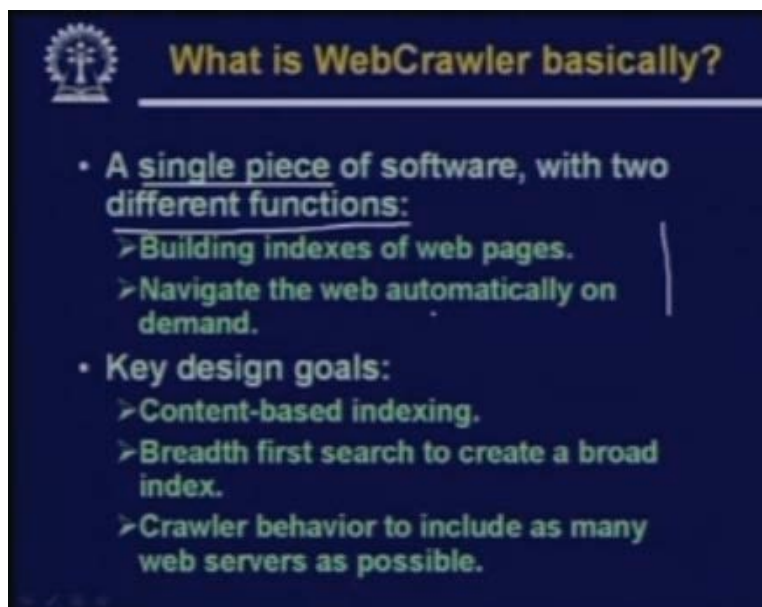


So let us see what web crawler is basically first. This I have mentioned already. WebCrawler is basically a tool that helps us in solving the resource discovery problem in the World Wide Web. Well the problem of search we are recasting it here as a resource discovery problem. Searching and resource discovery is similar. We are trying to discover some resources in the web where they are located. We are searching for them. So it is basically the same thing. Some of the feature so web crawler are like this. It

navigates the web automatically on demand. Now this demand can be driven by the website or by the search engine which is using web crawler for crawling the web.

So it has the capability of crawling or navigating the web automatically. This is important, I have mentioned in the last class because the web is changing very fast both in terms of its size and also in terms of the, you can say the life term of some of the pages in the web which changes very fast. So to perform the indexing what web crawler does it accesses the web documents one at a time sometimes taking decisions based on information available locally. Using this kind of information it tries to create the web index. So the principle is essentially the same. It has to go from one document to the next at any given time it will be looking at one particular document.

(Refer Slide Time: 04:30)



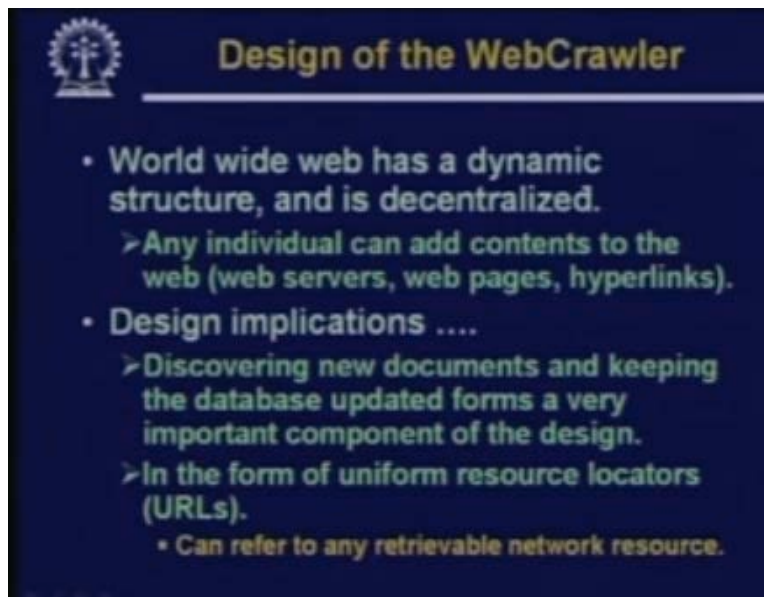
So basically the web crawler is an integrated piece of software. That is why it is called as single piece of software but with two different functions. The functions are firstly building index of the web pages. This is the most important function actually and secondly for the purpose of building indexes it has to go from one web document to the other. So navigation of the web in automated way this is also one of the main functions of the web crawlers. These are the two main functionalities. The key design goals of web crawler were that the indexing should be carried out based on the contents of the document, not based on the title or some name of the page like that. It is contents based indexing depending on or based on whatever is there in the body of the document indexing has to be carried out accordingly.

So what web crawler does it carries out breadth first search over the web to create a broad index. Now when we talk of this bread first search essentially the entire web is treated or is visualized as graph because we know in web we have so many pages. There are hyperlinks which are point to other pages. So if you treat the hyperlinks as edges in a

graph and the web servers as the nodes say essentially we are looking at a graph. So when we talk about going from one page to the other basically we are traversing the graph by following the edges from one node to the next. This is basic problem visualized or viewed as a graph theoretic problem. So the crawler behavior is utilized to include as many web servers as possible because this is one important thing. Why? Earlier we have said that the number of total web pages is huge.

A particular search engine may possibly cannot afford to index all of them. They have to consider a subset but which subset. This is one big question. So here what I was saying is that when we say that crawler behavior is in such a way to include as many servers as possible. Why? What we mean here is that when you say that are subset of pages must be fetched? We say there is subset should be searched that there should be one page one representative page from every web server. If there are 100 web servers there should be at least 100 pages one from each because when we get page from web server from that page some information about the other pages available locally can usually be found out. So this is the strategy which has been adapted by web crawler.

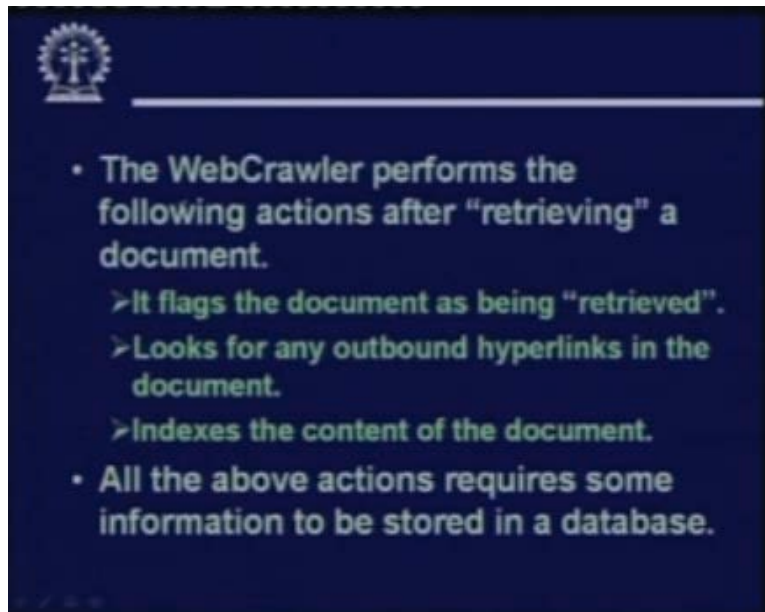
(Refer Slide Time: 07:52)



This you know world wide web has a dynamic structure. The creation of the web is decentralized. There is no one who has central control over the design of it. So any individual who has access to the internet can add contents to the web. So I can run a web server on my machine. I can create some web pages. I can provide hyperlinks to other pages no one can stop me from doing it. This is why we say that the World Wide Web has a dynamic structure. The structure again looking it as a graph; graph changes with time dynamically. Two nodes get added some pages get deleted means some nodes get removed and so on. So the design implications of this point is discovering new documents and keeping the database updated forms a very important component of the design.

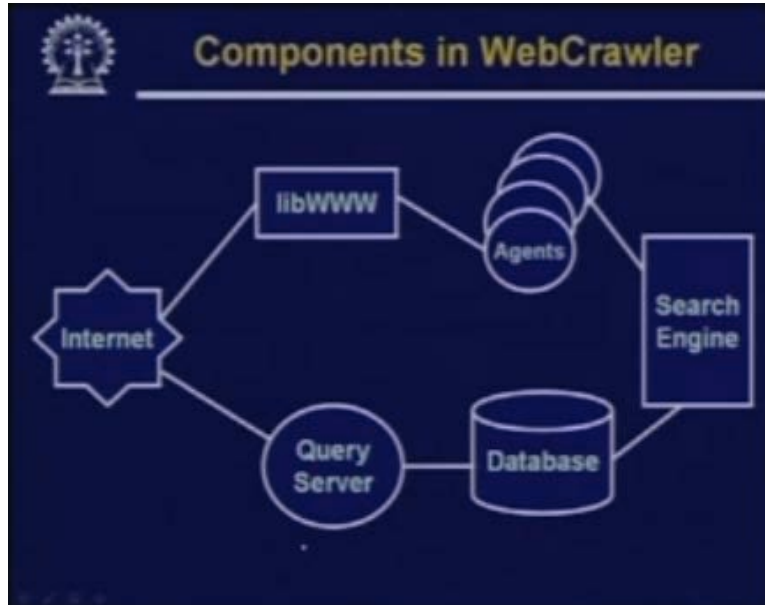
This is perhaps the most important issue because if you cannot do it in a good way your search engine cannot be that good. Because how good you are able to do the indexing this entirely will dictate how good your search engine will be. So discovering new documents obviously this is done by traversing uniform or universal resource locators URLs. So URLs are means of locating or going through a document for retrieving it. They in general can refer to any retrievable network resource. It can be another HTML document it can be an image it can be an audio clip, video clip or it can be also an executable file or some mail URL or some other kind of URL.

(Refer Slide Time: 10:08)



So after retrieving a document the web crawler will perform the following actions. Suppose the web crawler visited a page following a URL, retrieved it. Then it will carry out the following. It flags its own data base that the document in question has been retrieved. It looks for any outbound hyperlink in the document. Because if there are some hyperlinks going out of the document they may also have to traversed in the future and it looks at the contents of the document and carries out indexing. It indexes the contents of the document and all these above three operations they will require some information to be stored in a data base and of course access to the data base.

(Refer Slide Time: 11:19)



So pictorially the architecture of web crawler looks like this. We shall be looking at each of these individual components. We can see there is an internet out here. There is something called query server. Data base is of course there search engine agents and some library WWW. Let us see how these individual blocks in this diagram interact among themselves. And what are their respective functions?

(Refer Slide Time: 12:02)

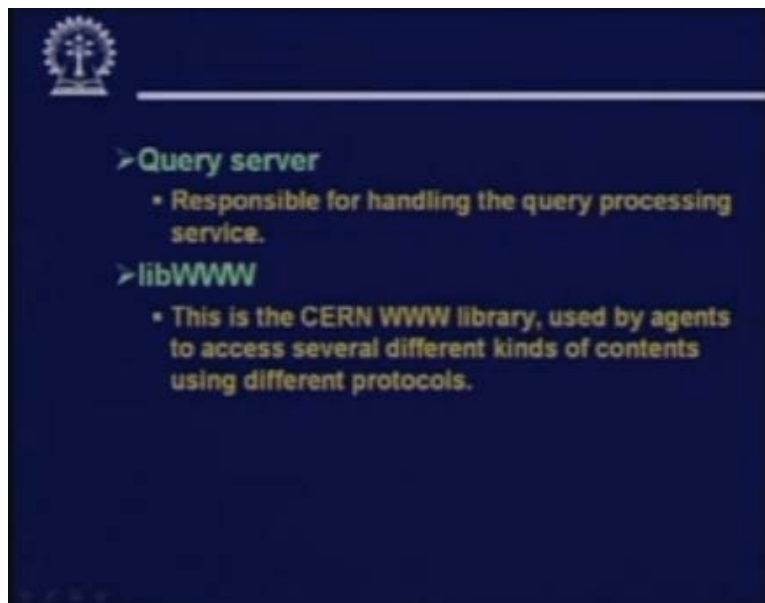
- The various components in WebCrawler.
 - Search engine
 - Responsible for deciding which new documents to explore, and for initiating the process of their retrieval.
 - Database
 - Used to store the document metadata, full-text index, and the hyperlinks between documents.
 - Agents
 - Responsible for retrieving the documents from the web under the control of search engine.

So the various components of the web crawler are as follows. Firstly we have a search engine. Search engine you can say this is a kind of a front end or the interface to the user.

The search engine will be responsible for deciding which new document to explore and for initiating the process of their retrieval. See here you can see that the search engine has two different roles depending on how or in what mode we are using the web crawler. Web crawler is run in two different modes one when it is trying to discover new documents. It is crawling the web and creating the index. Number two, when some user has a given a query it searches the data base for the index and looks for matches. So when it is the first one then the search engine will decide how to select the next document to explore. And when it is in the query processing mode it will initiate the process of their retrieval of the documents.

Next comes the database. Now in the data base as I said you need to store a number of different things you can store some information about the document. This is the document metadata. Document metadata can also contain keywords, full text index, can be separated also. Full index are the text based keywords based on which you have indexed the document. And of course the hyperlinks between documents from this document where else are the hyperlinks pointing to then you have agents. These agents are responsible for retrieving the documents from the web under the control of the search engine. So you just understand the role of the agent. So this agent is some kind of a tool which is running under the search engine. And it is the responsibility of the agent to retrieve the information requested from the web and give it back to the search engine. So agent also plays a very important role in this overall scenario.

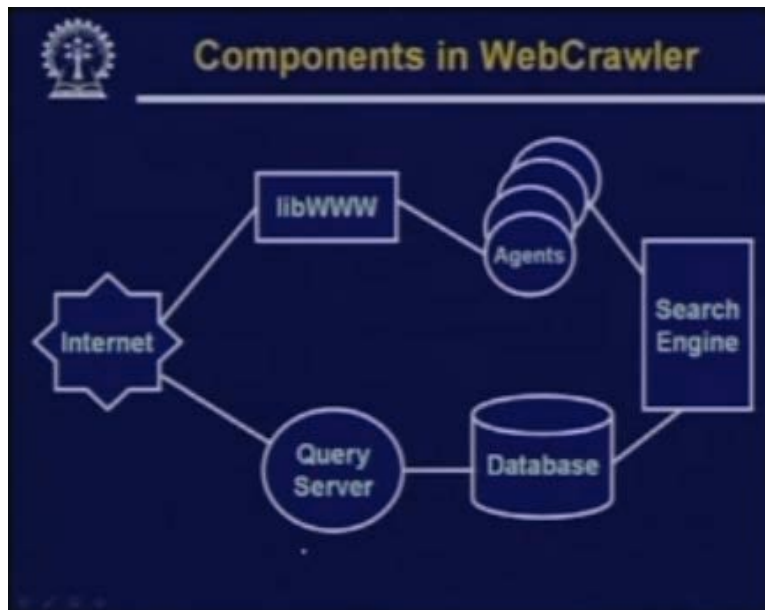
(Refer Slide Time: 14:51)



Then you have the query server which handles the user's queries. It is responsible for handling the query processing service. And lastly you have the lib WWW. This is exactly not a separate module or software rather this is a library. This is a standard library available. CERN WWW library. This library is used by the agents to access several different kinds of contents using several different protocols. Like this libWWW will

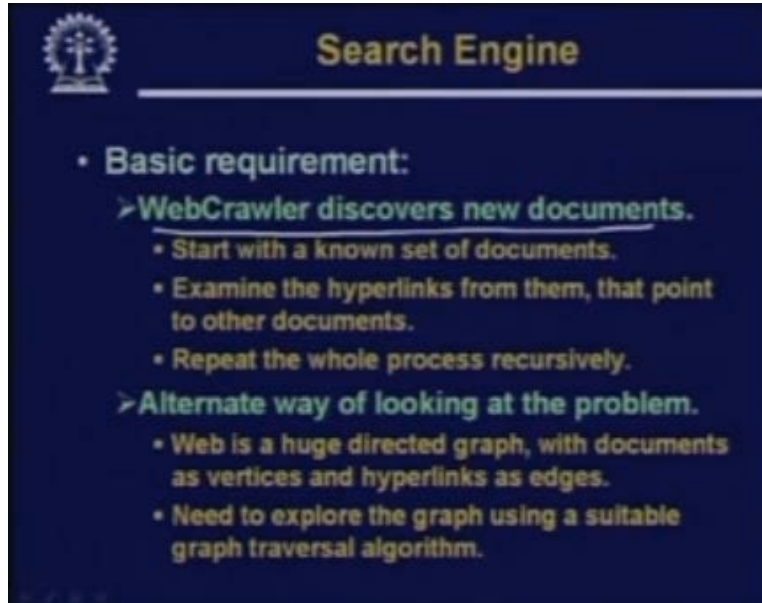
contain all the information about http, then the image documents whatever different protocols which may be running how to retrieve them buffer may also be there. So all these things are part of the libWWW package. So this libWWW package is basically used by agents before it can access the internet to retrieve the contents of the document that have been requested.

(Refer Slide Time: 16:08)



So just looking at the diagram once more now you can understand the role search engine. Search engine in one respect when it is trying to build up the index crawling, it will be taking help of the agents. Agents will be using this libWWW to search the internet. But when you are in the query processing mode the search engine will be building the data base. So the query server will be simply querying the data base. So you see that there are two different modes of operation here. One is while the index is being created and number two when the index is already there placed in the data base. The user has typed in some query and the queries are retrieved by matching against the indexes from the data base.

(Refer Slide Time: 17:16)

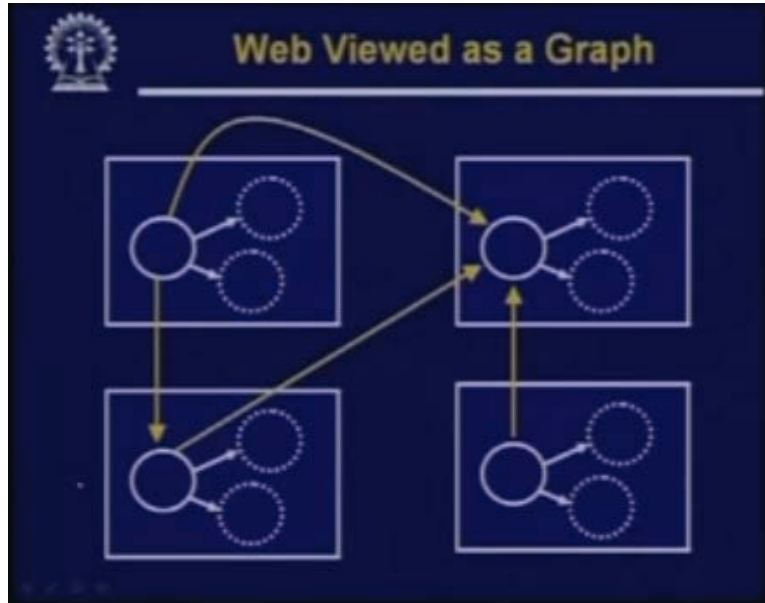
A slide titled "Search Engine" with a logo in the top left corner. The slide contains a bulleted list of requirements and an alternate way of looking at the problem.

- **Basic requirement:**
 - **WebCrawler discovers new documents.**
 - Start with a known set of documents.
 - Examine the hyperlinks from them, that point to other documents.
 - Repeat the whole process recursively.
 - **Alternate way of looking at the problem.**
 - Web is a huge directed graph, with documents as vertices and hyperlinks as edges.
 - Need to explore the graph using a suitable graph traversal algorithm.

So let us now look into the search in some detail. The basic requirement of the search engine is discovery of the new documents. Web crawler has to discover the new documents so how does it perform this step. Well web typically starts with a known set of documents. These URLs can be provided priorly from this known set of documents. It will examine all the hyperlinks that are pointing from them to other pages and will be continuing with the discovery following those hyperlinks. So this process will be repeated recursively. Recursively because whatever you are doing from the present page you will also be doing the same thing from the page you will be visiting by following the link.

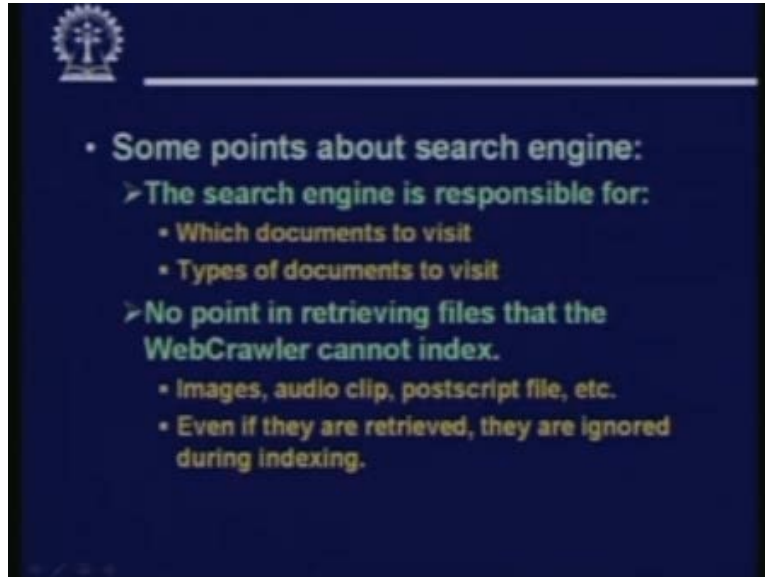
They will also have to look at what are the links to other pages, again you will be following them; again from there you will be doing the same thing so these are recursive processes. So as I said alternatively we can view the web as a graph. So the alternative way of looking into this problem is the web is regarded as a huge directed graph. The documents are the vertices nodes the hyperlinks are the edges. So basically the search problem now reduces to a graph traversal algorithm. So now we are forgetting web we are forgetting everything we are saying that. We have this graph and we will have to carry out some kind of a graph traversal using some criteria or fulfilling some you can say requirements.

(Refer Slide Time: 19:25)



So this diagram just shows you how we can visualize the web as a graph. These rectangular boxes are representative of the different web sites. So may be inside the same web site this particular document may be pointing to two other documents this and this. Similarly in some other website this document may be pointing to other documents. Now across web sites there may be pointers from this document it is pointing to this document. In this way you can have cross referencing. So overall you see that this is a graph like structure directed graph this is what we are talking about. So when we have this kind of graph like structure we will have to formulate our graph traversal algorithm in a proper way so that we know exactly what we are trying to do.

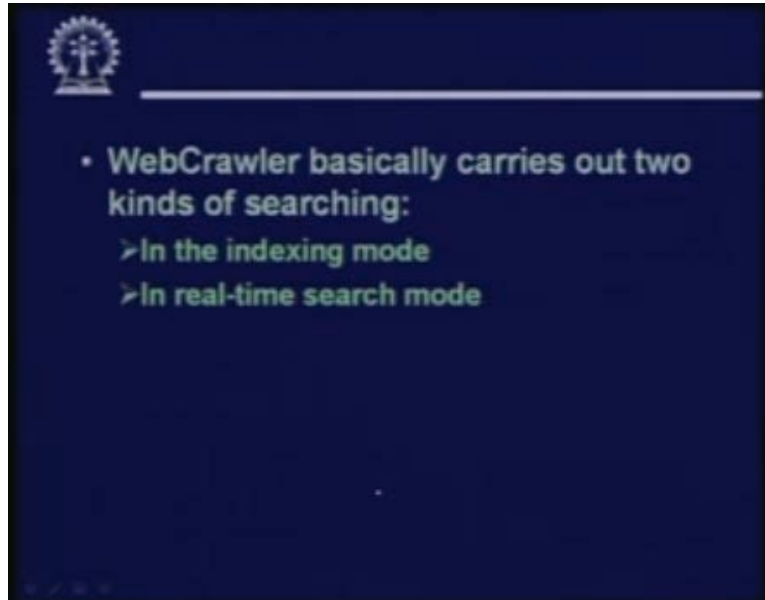
(Refer Slide Time: 20:20)



Some points here about the search engine. Now let us see that what are the main roles of this search engine? The search engine is responsible for which documents to visit and also types of documents to visit. See which documents to visit, is of course important. These we have said we can follow the hyperlinks. You can also have some additional criteria imposed. But it says it will also be responsible for the types of documents to visit because we are trying to understand our requirement. We are visiting the documents for the purpose of indexing. After visiting our document we shall be looking at the contents. We shall possibly be extracting some keywords and those keywords will go into the dictionary and this document will be said to be indexed after that. But suppose if I get some information from a website which is not such a readable document like it is an image, it is an audio clip, video clip, ps file.

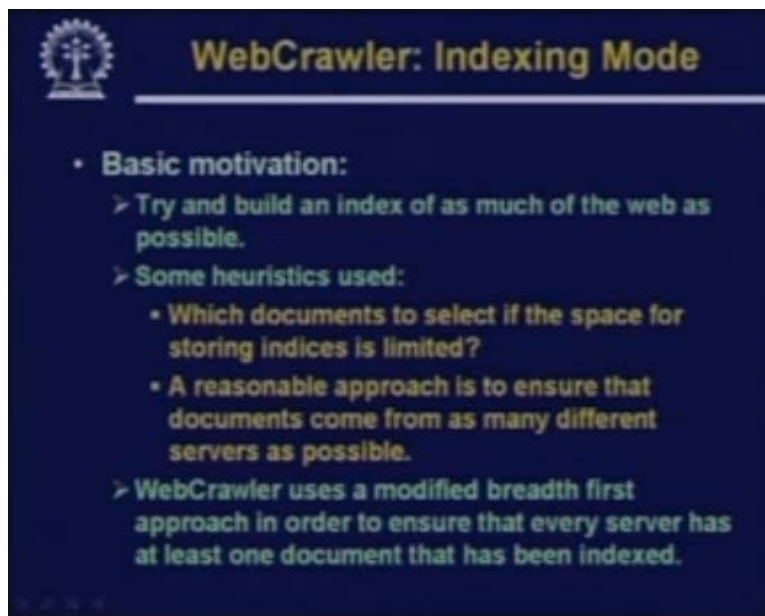
So there you cannot so easily evaluate or index the documents with respect to contents. If you are downloading an image you can say that this is an image file just by looking at the name of the file you can index with the name. But you really cannot do some. Indexing based on the contents you really do not know what is there inside that file. So from the point of view of indexing it is not justified to bring or retrieve any file which will not give you any additional information with respect to indexing. So web crawler follows this principle. It does not retrieve any file which it cannot index. No point in retrieving files that web crawler cannot index, this is the principal which it follows which it cannot index some examples I have already given. Images, audio clip, video clip, postscript file, even if you retrieve them by mistake during indexing they are simply ignored. Because you will not be looking at the contents in any case.

(Refer Slide Time: 23:01)



Now as I said that web crawler carries out searching in two different modes while it creates the index and while it responds to a user query. So these are the so called indexing mode and the real time search mode. So let us see how these two works.

(Refer Slide Time: 23:32)



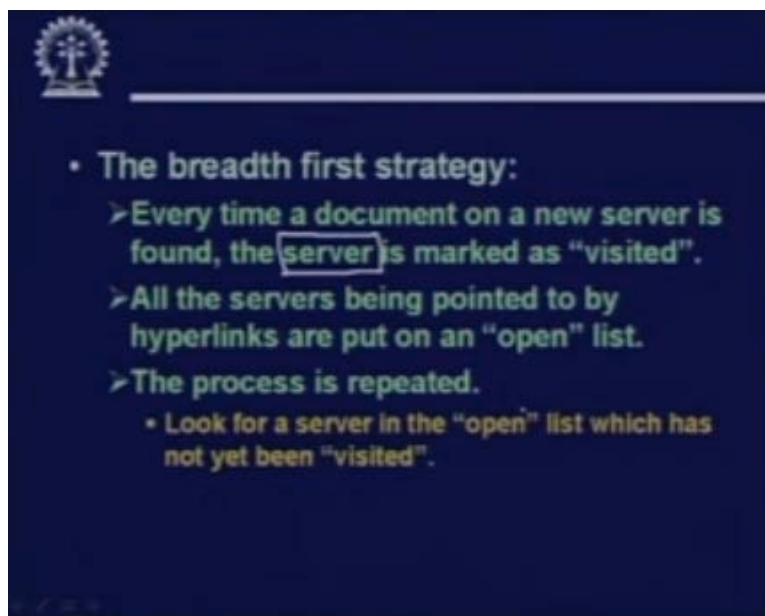
Now in the indexing mode as I said the main purpose is to crawl the web and create the index. Obviously the objective would be to index as much of the web as possible. But how the web crawler can ensure this? Let us see. So the basic motivation as I said it

should try and build an index of as much of the web as possible. This is a keyword, as much of the web as possible understanding. That it is not practically feasible to index the whole of the web. So web crawler uses some heuristics. This heuristics try to answer some questions which documents to select if the space for storing indices is limited.

Of course a reasonable approach as I mentioned earlier also is to ensure that documents come from as many servers as possible. Suppose you have space only to store 100 index values, make sure that those hundred index values come from 100 different websites, not 50 pages from the same website. This is the principle which the web crawler follows. So based on this criteria web crawler uses a modified version of breadth first search approach in order to ensure that every server has at least one document that has been indexed. So it is basically breadth first search but not across pages but across web sites.

So when it is exploring or crawling with respect to the hyper links it is also keeping track of the fact that which are the web sites which have already been explored. If your web site has been explored you just do not go there and explore it again. So if you do it. It will be just be a modified breadth first search. If the page is not visited but the website from where it is coming has been visited that is also considered as visited. This is the modification of the breadth first search. So using this modified version you can ensure that at least one page from every web site is fetched retrieved.

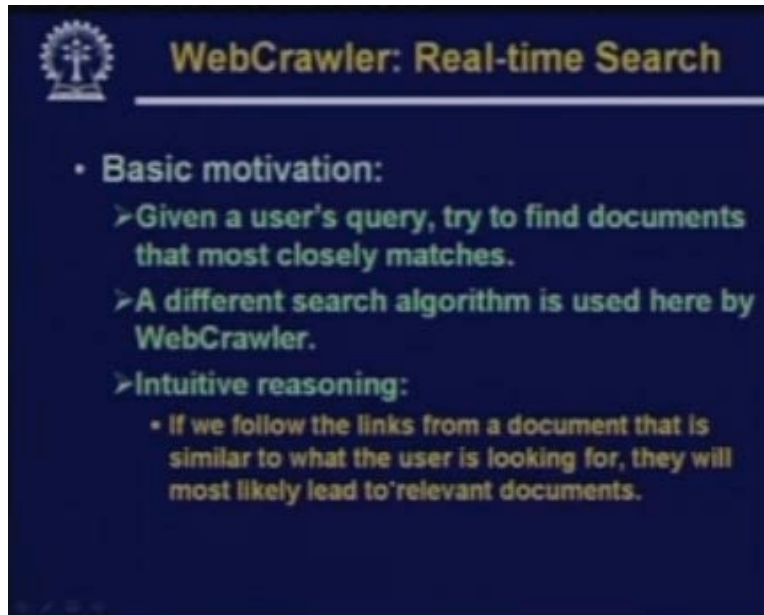
(Refer Slide Time: 26:04)



So it will be like this. Every time a document on a new server is found, the server is marked as visited not that the web document is marked as visited. You are marking the server as visited this is the keyword here. All the servers being pointed to by hyperlinks are put on an open list. The process is repeated, you look for the server in the open list which has not yet been visited. You open it and you again find the hyperlinks and find

out that whether they point out to web servers which had not yet visited. So this is the basic concept behind modified breadth first algorithm which is used here.

(Refer Slide Time: 26:52)



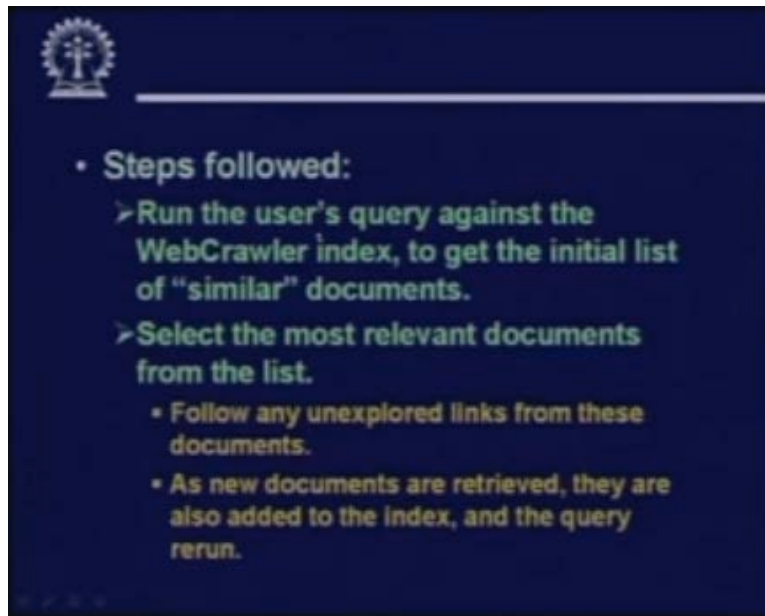
WebCrawler: Real-time Search

- **Basic motivation:**
 - > Given a user's query, try to find documents that most closely matches.
 - > A different search algorithm is used here by WebCrawler.
 - > **Intuitive reasoning:**
 - If we follow the links from a document that is similar to what the user is looking for, they will most likely lead to relevant documents.

And in the real time search mode this is the other mode I had mentioned. Here the web crawler is actually trying to answer the web queries. Given a user's query web crawler will try to find document that most closely matches with the query. So here of course the search algorithm will be different. See let me tell you one thing here. There are two issues here. On one side we are saying that we are not going to search the whole of the web. We are going to look at only a subset of the web and index them. But when you are actually searching for the documents during the query processing. There we may not like that some documents being excluded from the search. So during that time the documents where were not indexed in the first phase they may also be possibly explored. So this is something which you have to keep in mind. So in the query processing step the situation is not so simple it is not that the search engine simply looks at the index data base does some data base search and give you the result.

No, depending upon these keywords you have provided for search it can also carry out additional searches looking at additional web pages crawling and retrieve additional pages which will be also part of the search. So the search algorithm which is used by web crawler is a different one not the one which is used in the indexed mode. Intuitive reasoning of this different search algorithm is as follows. Well if we follow the links from a document that is similar to what the user is looking for they will most likely lead to the relevant document. Like what I mean to say is that, suppose the user is looking for flower and you have one document which is indexed already in the data base that is matched against the keyword flower. But if you look at the document you find that there are 10 hyperlinks going out to other pages. Most likely those ten hyperlinks will also be pointing to pages which have something to do with flower. So this is how the searching is done.

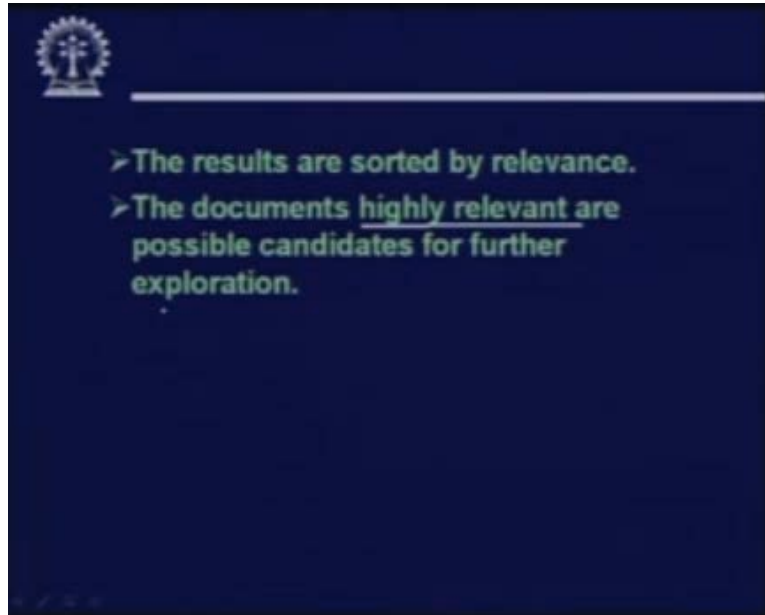
(Refer Slide Time: 29:45)



So steps followed are as follows. So the users query is first run against the web crawler index which means against the data base. So already in the data base we have a large number of documents which have been scanned and indexed. So when you run the query and search the data base you will get the initial list of documents which are considered to be similar with respect to the search keywords. Now based on some way of measuring relevancy relevance you select the most relevant documents from the list which is returned from the data base. You look at these documents from the list, follow any unexplored links from these documents. Because many of these documents are not indexed. So you look at those documents which have come out of the index and follow the hyperlinks to bring those documents which are not yet being indexed.

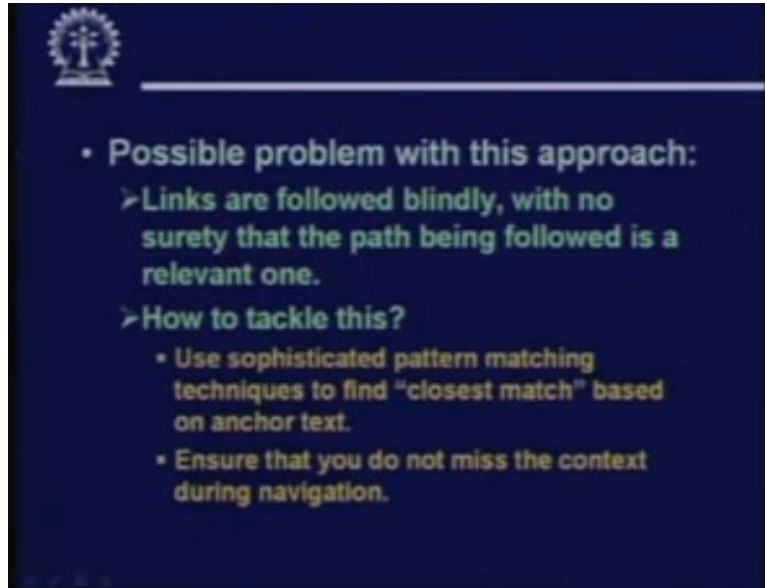
So as the new documents are being retrieved , they are also added to the index and query is re run. This is done dynamically, based on a query you are possibly retrieving some new pages repeating the process of indexing and you are going on updating the data base. So the idea is that proactively you do not try to visit all the pages and start indexing. You start by an initial set of index where there will be at least one representative from each web server. Then, depending on the user queries you proactively go to other pages and try to build up the index during that time dynamically. So this in general will lead to exclusion of a huge number of pages which otherwise or not that useful or not that sort after.

(Refer Slide Time: 31:57)



So the results which are returning out here. So they are sorted by relevance and the documents which have maximum relevance value they will be the next possible candidates for further exploration. So in this way you go on recursively expanding documents going from one to the other. So here you find that here web crawler have introduced one interesting concept in the sense that it is not purely two part process. In what part you create the index, in other part you search it. No, during searching also if you do not find that the index is there you proactively go to other pages and try to expand the size or the capability of your index. This is something which was worked quite well with this web crawler and many of the subsequent web crawler based engines have incorporated such concepts there.

(Refer Slide Time: 33:02)



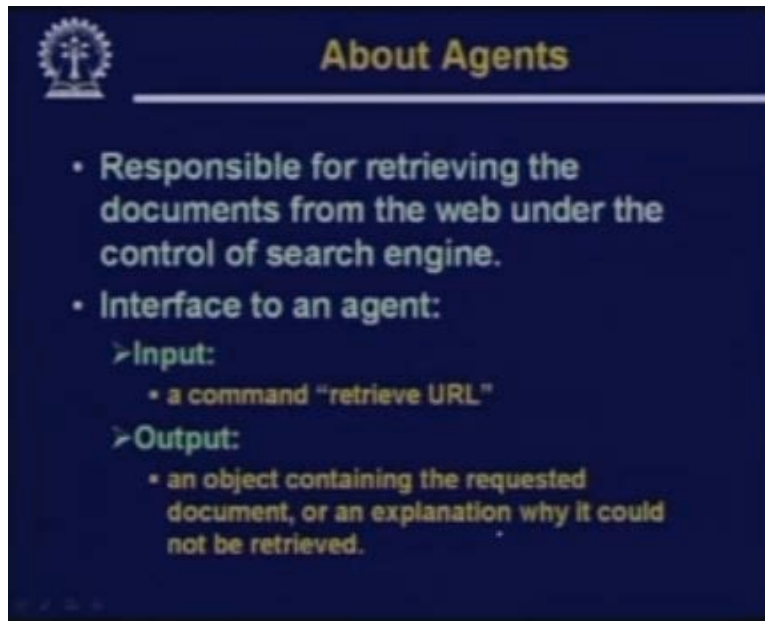
There are of course some possible problems here also. This kind of a so called blind approach. Here when we are considering one document then all the links which are going out of a document are followed blindly with no surety that the path being followed is a relevant one. Like for example I have a document which talks about flowers which is my current document. There are ten links from there. Now one of the links may be pointing to my own home page. There can be another link which may be pointing to some encyclopedia site. There may be another link which may point to some Google or some website which I have put it in my page.

So just by looking at the hyperlinks you cannot know that which of the hyperlinks can possibly lead to a page which is relevant in the present context. So since the links are followed blindly, there is no surety or guarantee that the path being followed will lead you to a page which is relevant. Now how to tackle this? Tackling this requires again some sophisticated techniques. You use sophisticated pattern matching techniques to find closest match based on anchor text. So here the concept of anchor text comes in recall. What is an anchor text? In an html document you can specify a hyperlink. How you can specify a phrase or a block of code as an anchor and the entire anchor you can specify as the clickable hyperlink?

So when the document is being displayed on the screen, if someone clicks on the text that new URL will be retrieved, so anchor is the text or whatever which the user has to click to go to that hyperlink. Now here what web crawler says is that you better look at what that anchor says. Just do not look at the hyperlink. The anchor might say blue rows or pink rows. Blue rows are a rare variety. So if you click on that possibly you will be going to a page which talks about that. So you give more weightage to what is there in the anchor text, do some careful natural language processing on that. So processing of anchor text can provide quite relevant and important information.

This will also ensure that you do not miss the context during navigation because if you do not do this what may happen very soon. Starting from a page containing flower you have moved on to a set of pages which are talking about everything else other than flowers. So you have shifted context to something else. This means if you do it blindly there is a definite chance of that happening. But if you do it in a systematic way at every stage if you give importance to the anchors if you see how they are how they are worded you try to understand the meaning and try to follow the links. So this will also ensure that you are within your range or domain of interest.

(Refer Slide Time: 36:55)

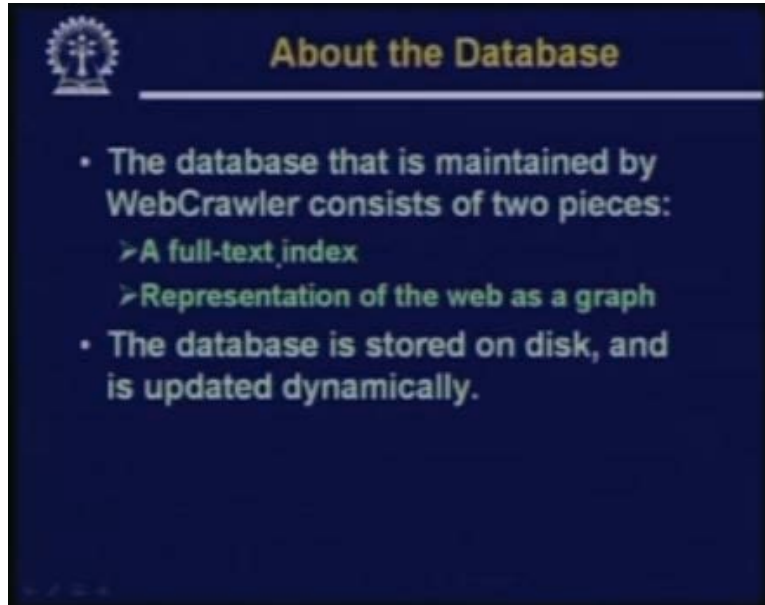


The slide, titled "About Agents", features a logo in the top left corner. The main content is a bulleted list:

- Responsible for retrieving the documents from the web under the control of search engine.
- Interface to an agent:
 - >Input:
 - a command "retrieve URL"
 - >Output:
 - an object containing the requested document, or an explanation why it could not be retrieved.

Now talking about the agents which are there as part of the web crawler architecture agents we had mentioned. They are some software code utilities or processes you can say. They are responsible for retrieving the documents from the web under the control of the search engine. So the search engine can find out that there are four hyperlinks going out. So the search engine can ask the agents that well these are the four documents I need to retrieve please do it. So in orders to do these things in parallel possibly in many cases several concurrent versions of agents are formed may be as processors or as threads. So the interface to an agent will be simple. The search engine will simply give it a command like retrieve a given URL. So if you are an agent I will simply tell you that. Well, look this is a URL I want you to retrieve the URL give it back to me. This is so the interface is very simple. Input and output what you will be giving back to me is an object that will contain the requested document. But sometimes the agent may exercise some intelligence that it will not retrieve the document because it may feel that it is either not present. It is unable to locate the document on the particular web server or it feels that the document is not of relevance in the present context. So some local intelligence can also be dedicated to the agent, so the agent can also filter some pages from being forwarded back to the search engine.

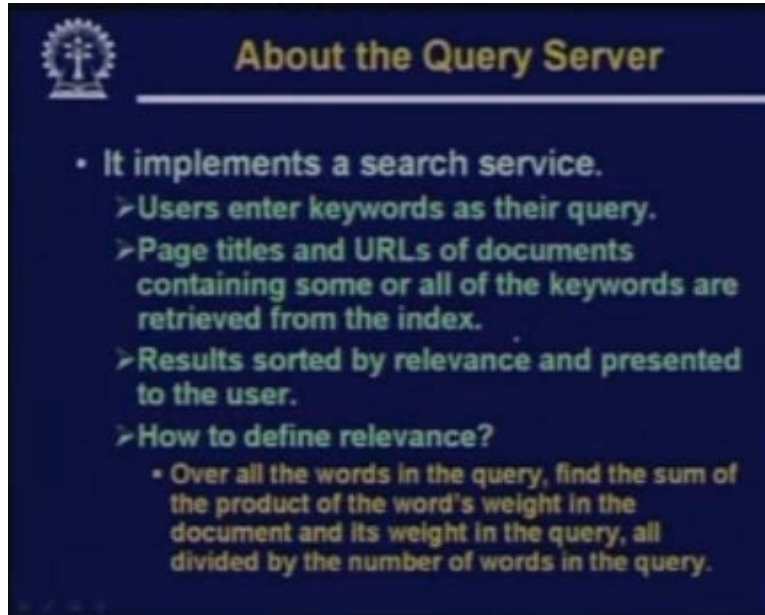
(Refer Slide Time: 38:52)



Talking about the data base. Data base, there are two parts to it. Full text index. Full text index are basically the keywords against which search results may be provided and representation of the web as a graph. See representation of the web as a graph it can be implicit it can be explicit. If the data base contains all the pages that have been indexed they by default contents the hyperlinks inside them. But for fast searching information about the hyperlinks can be stored separately which essentially means I have stored information about the graph storing the pages and information about the hyperlinks essentially means it is some kinds of an adjacency list.

I am storing information about the vertices and the nodes which are adjacent to it. So the data base which comprises of these two things are stored on disk and obviously will be updated dynamically. Both during the creation of the index and also during the searching query processing. Next let us come to the query server. Now the query server is the module which accepts the user's queries. Typically as I said that the users will be entering keywords as their queries. For example for searching for that Asian test symposium someone I may just enter the keyword test symposium or ATS or ATS 2005 something like that. So depending on the user the keywords provided may be quite detailed or may be quite sketchy.

(Refer Slide Time: 41:00)



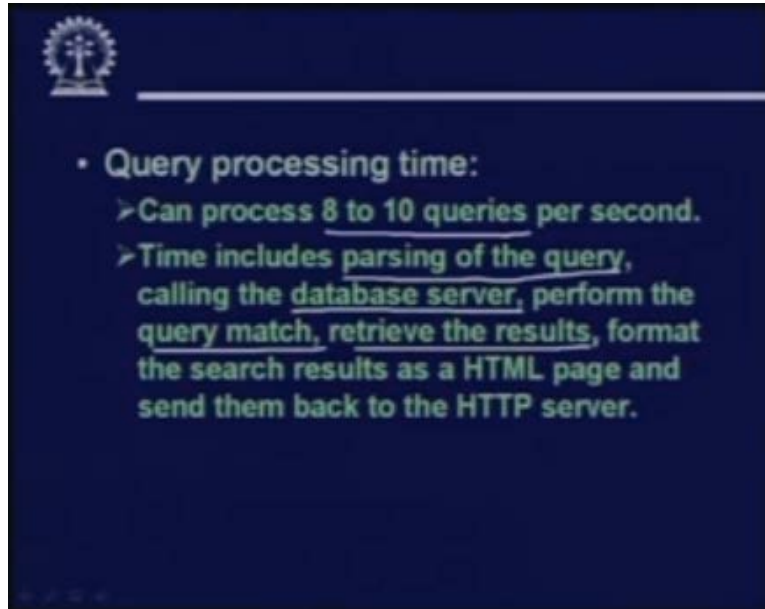
About the Query Server

- It implements a search service.
 - Users enter keywords as their query.
 - Page titles and URLs of documents containing some or all of the keywords are retrieved from the index.
 - Results sorted by relevance and presented to the user.
 - How to define relevance?
 - Over all the words in the query, find the sum of the product of the word's weight in the document and its weight in the query, all divided by the number of words in the query.

So what the query server will do? In the second step it will be retrieving information that is stored in the data base in the form of the full text index values, page titles and the URLs of documents containing some or all of the keywords are retrieved from the index. Now after retrieving this information results are sorted by relevance. So there are again number of different ways some simple some quite complex which can be used to measure relevance. And after sorting the result they are presented to the user so that the most relevant user relevant result will appear first because out of 10000 results if they all appear in a random order then which are the most relevant ones you may have to put in a lot of effort to find that only. So they are sorted by relevance typically. So how to define relevance. Well a simple heuristic is used here. Well over all the words in the query like for example I have given test symposium 2005. There are three words. Find the sum of the products of the words weight in the document and its weight in the query.

See here there are two terms we have introduced weight of a word in a query. Weight of a word in a document. Now in a query very rarely we type in an English question. Some keywords we type in order that comes to our mind. So most likely the word that we have typed in first is the most relevant one. Next one is the next relevant. So the weight of the word in a query can very easily be determined by position in the query string. Similarly the weight of a word in a document can be judged using some heuristics. As we have mentioned in our last class, like whether the word appears in the title; whether it appears within a sentence or a paragraph towards the beginning of the document and so on. These are a few measures you can use or combine to provide or produce some kind of a quantitative estimate. So after finding the sum of the product of the words weight and its weight in the query you divide it by the number of total word. Total number of words in the query. So this is how the query server of web crawler determines a numerical relevance figure when it is trying to solve the results.

(Refer Slide Time: 44:10)



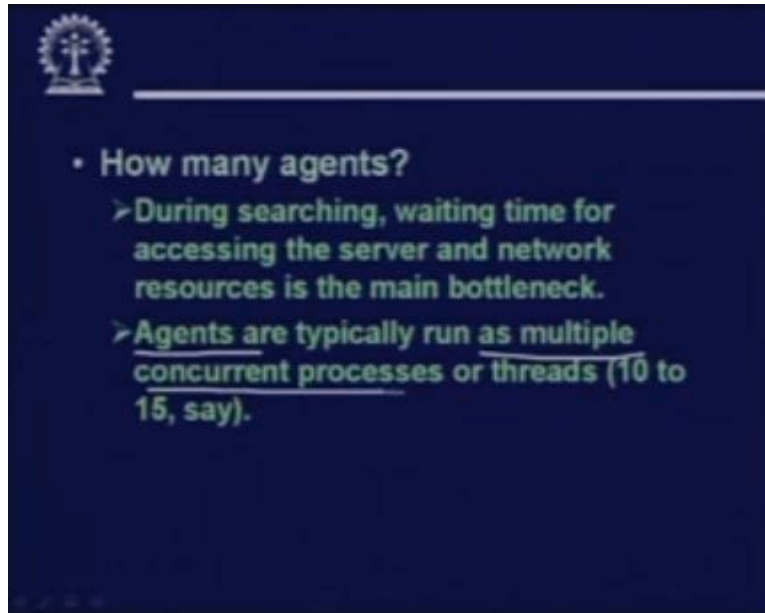
Now talking about the query processing time, typical bench marking has shown that it was able to process 8 to 10 queries per second. Of course if the queries are very complex it may require more time. But these are very typical queries that have been bench marked. So this processing time we are talking about here this consists of number of different components. The query has to be parsed; there can be just simple words there can be codes there can be other special characters and are not connectives so on. So the queries has to be parsed. You will have to invoke the data base server to search for the index values. Perform the query match; retrieve the results from the data base. Well of course ranking. Ranking by relevance would also come here.

Then it will also have to format the search results as a HTML page. Because all these processing are being done by these server this server side. But the user is still sitting on the browser as the client. So it has to come back to the user as a nicely formatted HTML page. So after doing all these things it has to be formatted in the form of an HTML page in a nicely formatted way. So I will send them back to the HTML HTTP server. So this is how web crawler works. Now you see with this idea it will not be difficult for you to develop your own search engine. Let us try to see what are the things you need to know in order to develop a search engine from scratch. First and foremost you need to know how to retrieve page from a web server given its URL. That is the standard HTTP get command. So using the get command you can retrieve a document or the post command.

So that part is done given a URL possibly retrieve a document. Second thing is that you have to maintain the data base. So as you are scanning through the web as you are scanning through the links you will be building your own your own information base in terms of the index structure dictionary. So once you have dictionary in place there will be another layer on top of it which will talk about the query processing. The user will be typing some query at the back end the queries will be processed and the index values will

be searched and whatever algorithm you follow you can follow something like alta, vista or you can follow something else also. But basically these are the few different things that you will be requiring to do that.

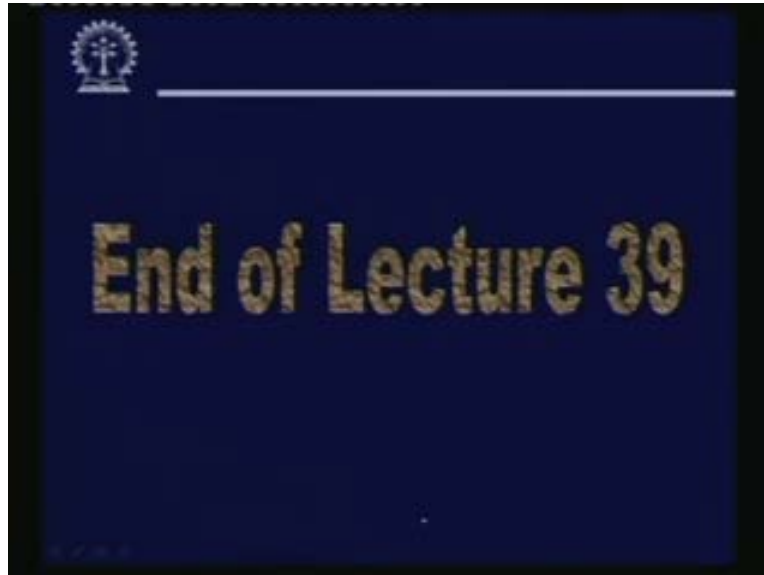
(Refer Slide Time: 47:38)



So with that we come to the issue of means how many agents are required because see we have talked about so many different things. We have talked about the query server, the search engine software. We have talked about the data base but how many agents are required. Agents I mentioned once when we talked about the search engine. That search engines will be sending out requests for retrieving pages given the URLs depending on the number of links that are going out from a page. Now as I said it will be faster if this process can be started together concurrently, so what web crawler does? Web crawler runs agents as multiple concurrent processes or threads.

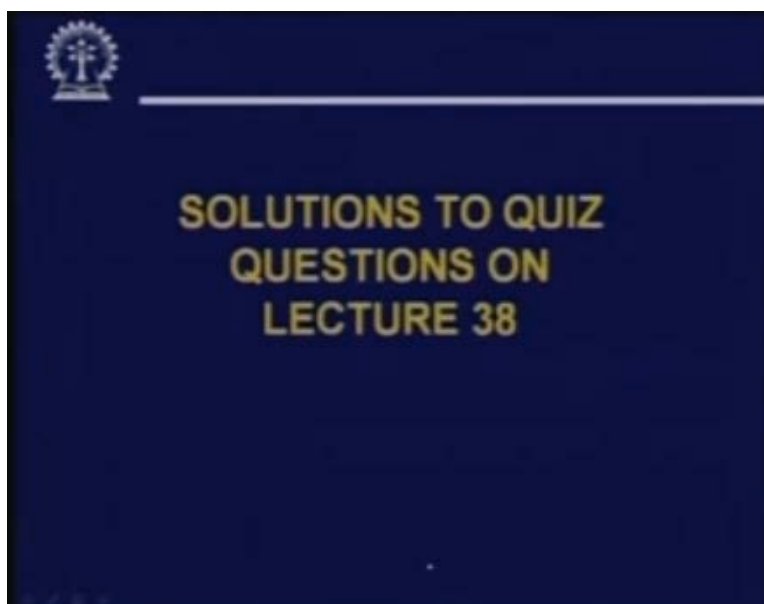
Well it depends on the implementation typically up to 10 or 15 search processes or threads are run. This is actually used to circumvent the waiting time for accessing server network resource because the thing is that if the search engine is trying to build up the index sequentially one after the other instead of spawning concurrent agents. What might happen is that one particular page it is trying to retrieve due to some network congestion it is taking considerable amount of time for the page to be downloaded. So during this entire duration the search engine will be lying idle. But if it is done concurrently may be some other pages have come in the mean-time. So it can start processing them. So this is the basic idea.

(Refer Slide Time:49:36)



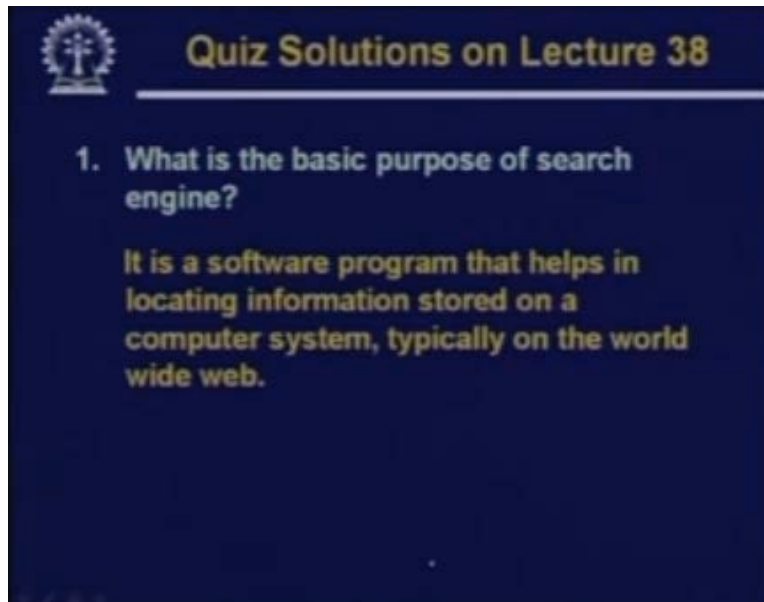
So with this we have to the end of our lecture number 39. Now here basically as I said we had talked about search engine works with specific case study web crawler. These are very interesting implementation of web crawler. So how web crawler works? What are the basic concepts and strategies web crawler uses and some heuristics also we have talked about? So with this background I believe you can take up a project of developing your own web server. If you want to but my objective in this class was to give you some underlying understanding of how web server works. So that you know how to go about the implementation if it is required at some point in time for you in the future.

(Refer Slide Time:50:33)



So let us now move on to the solutions to the quiz questions of the last class.

(Refer Slide Time: 50:44)



The slide features a dark blue background with a white logo in the top left corner. The title "Quiz Solutions on Lecture 38" is written in yellow at the top. Below the title, the first quiz question is listed in white text, followed by its answer in yellow text.

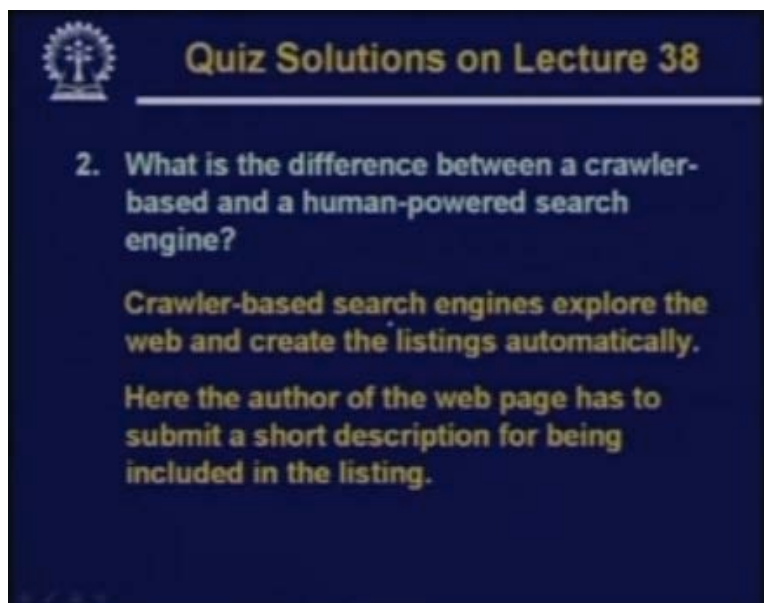
1. What is the basic purpose of search engine?

It is a software program that helps in locating information stored on a computer system, typically on the world wide web.

The first question was what is the basic purpose of search engine?

Well search engine this I have said repeatedly it is a software program that helps in locating information stored on a computer system typically on web servers world wide web of course.

(Refer Slide Time: 51:10)



The slide features a dark blue background with a white logo in the top left corner. The title "Quiz Solutions on Lecture 38" is written in yellow at the top. Below the title, the second quiz question is listed in white text, followed by its answer in yellow text.

2. What is the difference between a crawler-based and a human-powered search engine?

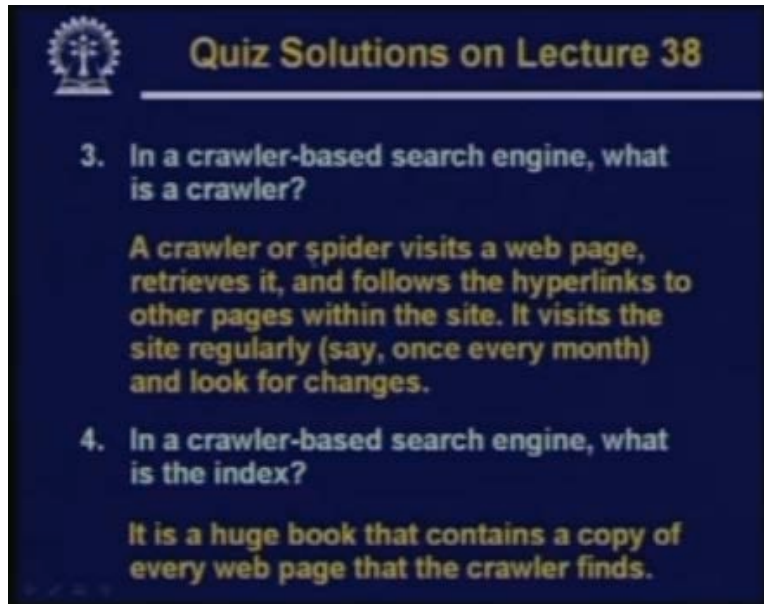
Crawler-based search engines explore the web and create the listings automatically.

Here the author of the web page has to submit a short description for being included in the listing.

What is the difference between a crawler based and a human powered search engine?

Now a crawler based search engine as I said they will go from one website to another automatically they will be exploring the web and create the listings and the index dictionary index automatically. However in a human powered engine here the author of the web page has to submit a short description for being included in the listing. So as long as the author does not submit it. The corresponding page will not be indexed. So for the purpose of indexing we will have to submit the keyword and other information description about your website.

(Refer Slide Time: 52:02)



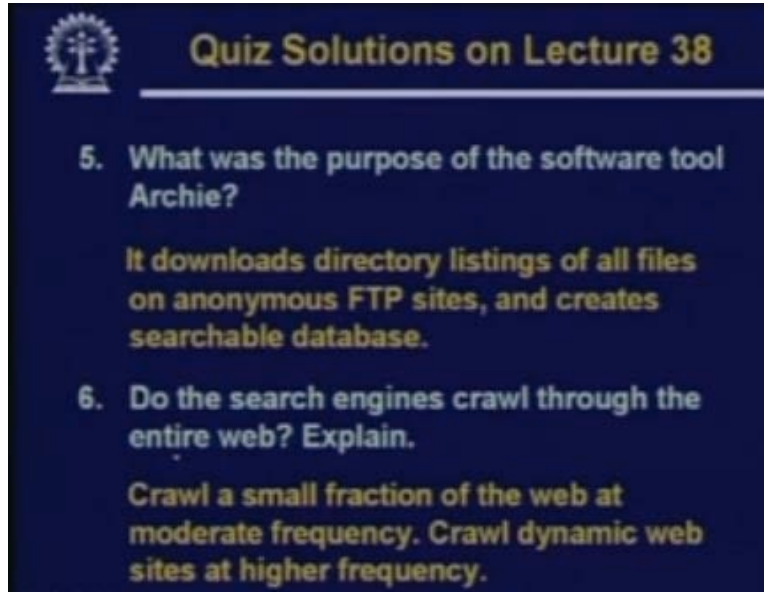
In a crawler based search engine, what is a crawler?

A crawler is also called a spider. The main purpose of it is to visit a web page retrieve the web page. This can be done using the http protocol and after retrieving it can look at the contents and find out the hyperlink. It will follow the hyperlinks that are present in the document to other pages within the site. Now it visits the site regularly as I said it can be once a month it can be more frequent also and look for changes if there changes they are again brought they are again downloaded.

So in a crawler based search engine what is the index?

Well index is a huge book or a dictionary that contains a copy of every web page that a crawler finds. This is used during the time the search or the query processing is done.

(Refer Slide Time: 53:10)



The image shows a slide titled "Quiz Solutions on Lecture 38" with a logo on the left. It contains two quiz questions and their solutions. Question 5 asks about the purpose of the software tool Archie, and the solution states it downloads directory listings of all files on anonymous FTP sites and creates a searchable database. Question 6 asks if search engines crawl the entire web, and the solution explains they crawl a small fraction at moderate frequency, with dynamic sites crawled more frequently.

What was the purpose of the software tool Archie?

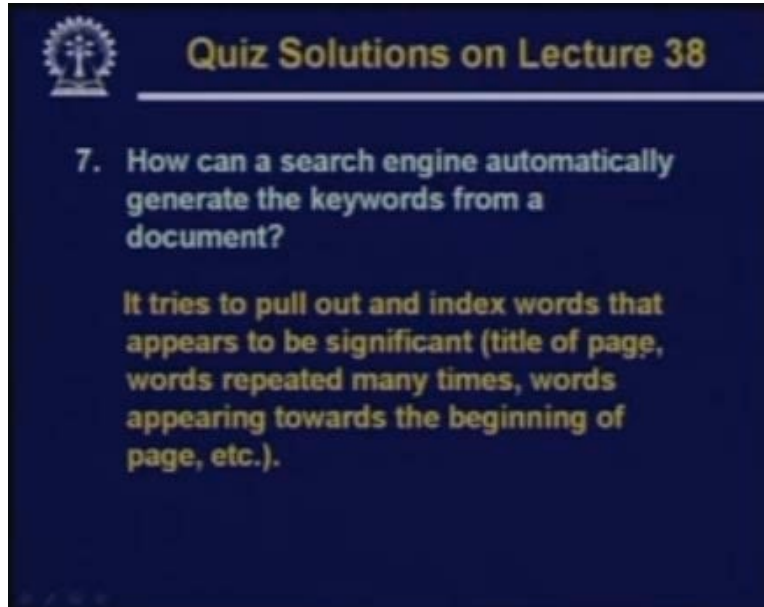
Well Archie was used to download directory listings of all files on anonymous FTP sites and create searchable data base so that later on when users submit it as queries based on some keywords. It would search the data base and will give you a listing of the anonymous web sites where possibly the information you are requesting for will be found.

Do the search engines crawl through the entire web explain?

Now I had mentioned because of the shear size of the web it is not possible.

So search engines like that web crawler we have seen in the design they crawl small fraction of the web at moderate frequency. Blocks new sites which change very frequently they are crawled at higher frequency. You should make a distinction that which of the sites are blocked sites are such new sites.

(Refer Slide Time: 54:14)



The image shows a slide titled "Quiz Solutions on Lecture 38" with a logo in the top left corner. The slide contains a question and its solution.

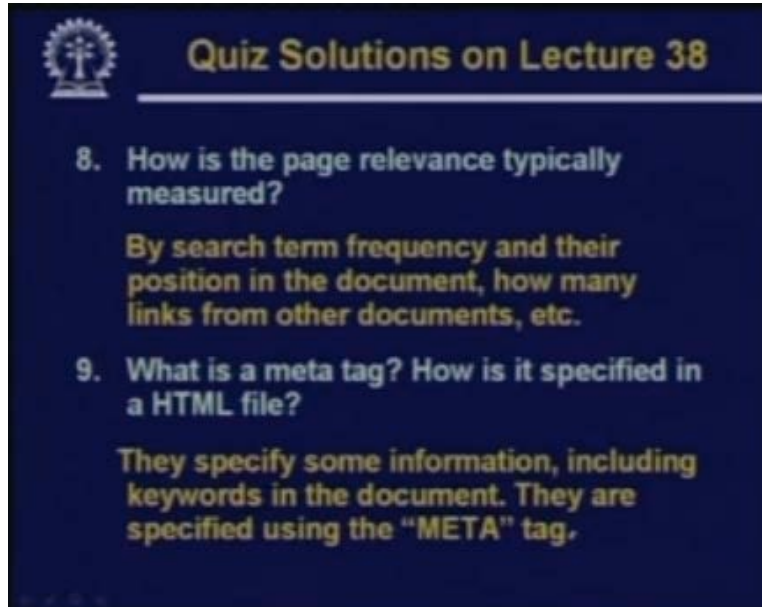
7. How can a search engine automatically generate the keywords from a document?

It tries to pull out and index words that appears to be significant (title of page, words repeated many times, words appearing towards the beginning of page, etc.).

How can a search engine automatically generate the keywords from a document?

Well we have mentioned that the search engine can extract words from the document and index them depending on their significance like consider a several heuristics here. So I am repeating the title of the page. So the words appearing there can be considered to be significant words repeated many times they may be considered to be significant. Those appearing towards the beginning of the page they may be more relevant as compared to those which are appearing towards the end. So these are some heuristics you can use to just automatically extract some useful information about a document without seeing it in front of you.

(Refer Slide Time: 55:03)



Quiz Solutions on Lecture 38

8. How is the page relevance typically measured?

By search term frequency and their position in the document, how many links from other documents, etc.

9. What is a meta tag? How is it specified in a HTML file?

They specify some information, including keywords in the document. They are specified using the "META" tag.

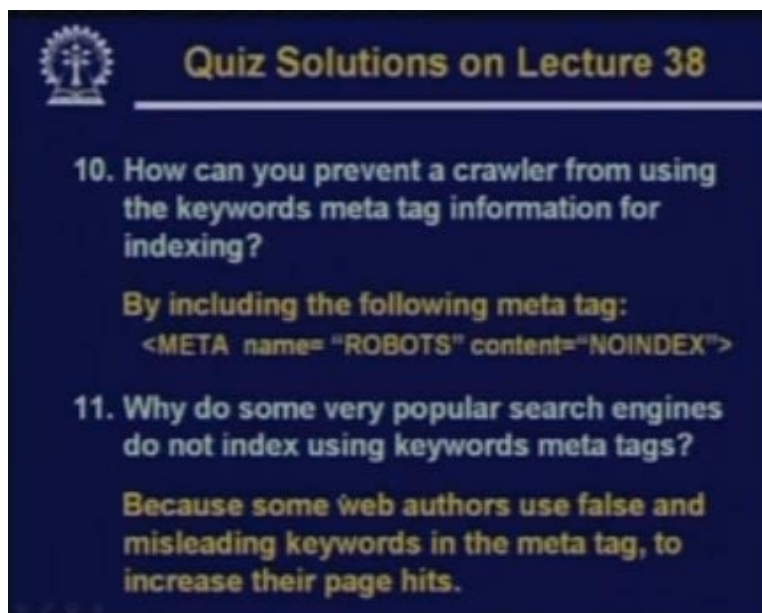
How is the page relevance typically measured?

By search term frequency and their position in the document this is one and how many links are pointing to it from other places. These are some typical measures.

What is a Meta tag? How is it specified in a HTML file?

Meta tags actually they specify some information including keywords in the document. They are specified using the Meta tag.

(Refer Slide Time: 55:42)



Quiz Solutions on Lecture 38

10. How can you prevent a crawler from using the keywords meta tag information for indexing?

By including the following meta tag:
`<META name= "ROBOTS" content="NOINDEX">`

11. Why do some very popular search engines do not index using keywords meta tags?

Because some web authors use false and misleading keywords in the meta tag, to increase their page hits.

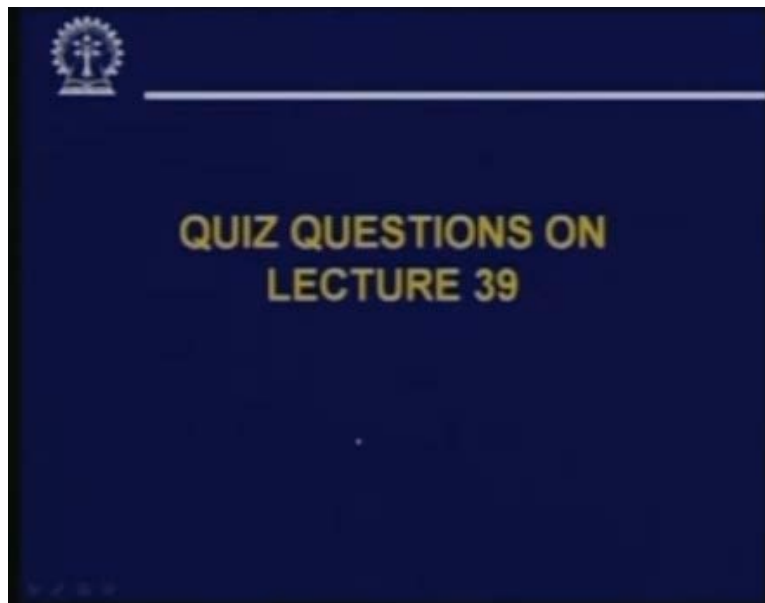
How can you prevent a crawler from using the keywords Meta tag information for indexing?

If well you just include this line in your HTML document this will prevent the web crawler from indexing your site. META name equal to ROBOTS content equal to NOINDEX.

Why do some very popular search engines do not index using keywords Meta tags?

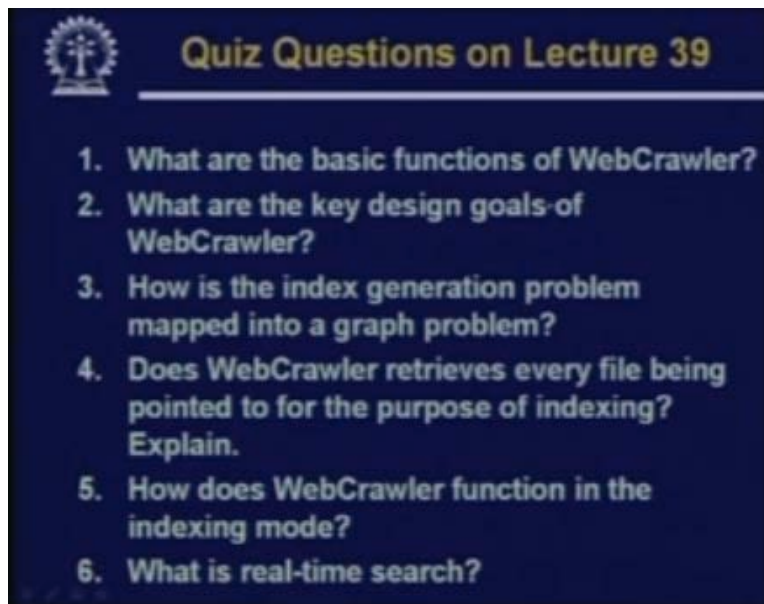
Because we have mentioned that there are many web authors we use false and misleading keywords in the Meta tags for the sole purpose to increase their page hits. See there are many websites where there are lot of advertisements and they gain revenue the number of hits number of peoples that visit the websites. Just to increase the revenue they can include very popular keywords in their Meta tags so that many people visit their page by accident or otherwise. So they can generate they can earn more amount of money out of their advertisement and other ventures.

(Refer Slide Time: 57:00)



Now some questions from today's lecture.

(Refer Slide Time: 57:04)



What are the basic functions of WebCrawler?

What are the key design goals of WebCrawler?

How is the index generation problem mapped into a graph problem?

Does WebCrawler retrieve every file being pointed to for the purpose of indexing?

Explain.

How does WebCrawler function in the indexing?

What is real time search?

So we are almost near the end of this course. Today's lecture was the 39th one. In the next lecture we shall be concluding our course by providing a summary of whatever we have covered so far. And also I will try to give you a few pointers to further studies and further explorations. This should be considered as just a platform from where you can explore new things because this internet is a vast and a very fast changing technology. So unless you keep yourself updated you will very soon become absolute. So we shall be talking about these issues in our next and last lecture. Thank you.