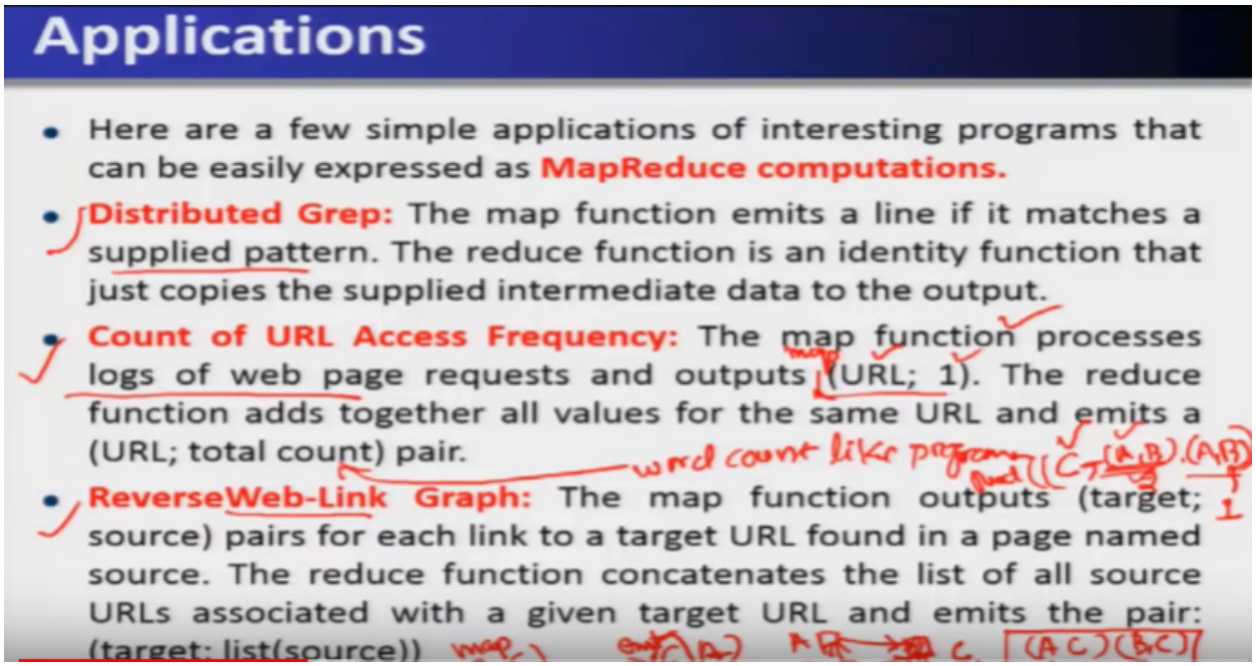
Lec 07-

Hadoop MapReduce 2.0

(Part-II)

Let us see, some of the applications, of MapReduce.

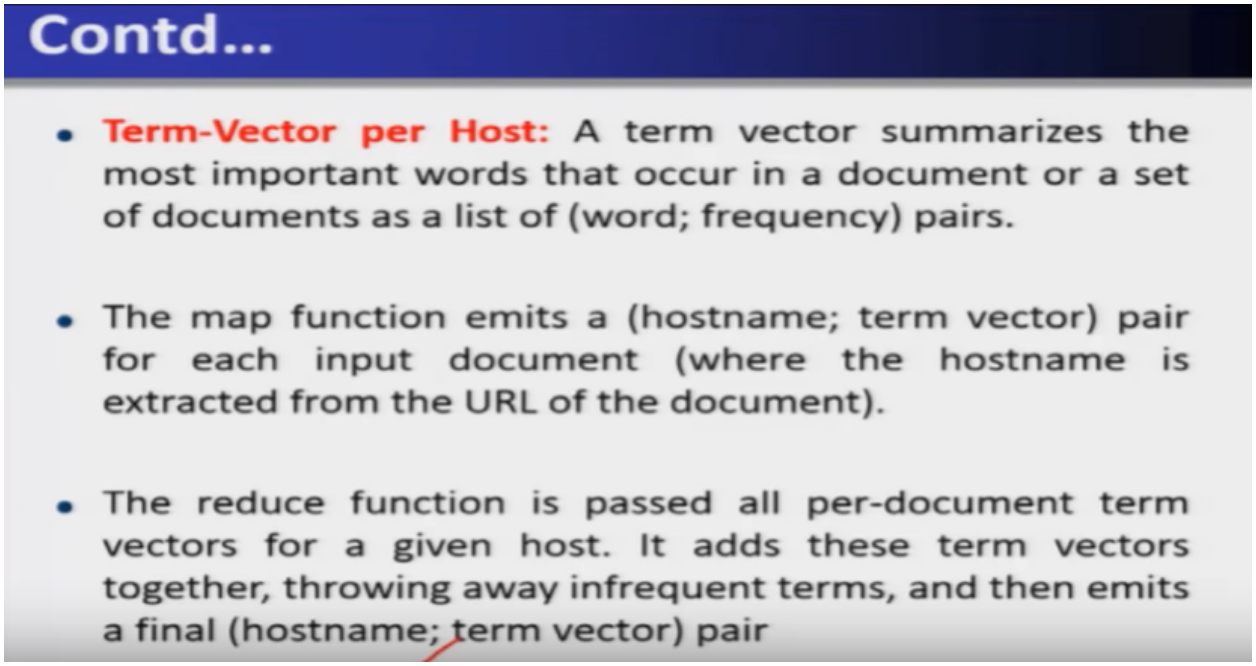
Refer Slide Time :(0: 15)



We are already the programs are available and its, use in the production, environment, by different companies. So, here are the few, simple applications of interesting programs, that can be, that has, already been expressed, as the MapReduce computation. So, the first one is called, ‘Distributed Graph’. Here, the map function emits a line, if it matches a supplied pattern, for example, if the document is given and we, we have already given one pattern. So, all the lines in that particular document, where that particular pattern is appearing, will be emitted and reduced function will become an identity function that that just copies, the supplied intermediate values to the output that is called, ‘Intermediate Graph’ or ‘Distributed Graph’. so, the difference between the Graph and this distributed graph is that, here the document can be very big, it cannot fit into one system, memory and therefore a document which is distributed the stored on data nodes, can perform this operation the graph. So, it will filter and extract, only those set of documents where we are interested in that particular pattern, that's called, ‘Distributed Graph’, has various purposes various applications, the next application is the count of URL, access frequency. Now, to do this, there is a map function, which processes the log of webpage, requests and the output URL with the value 1. So, that means the map function, what it does is? It inspects the log of web pages and for every URL, it encounters it will emit one as per the map phase. Now, reduce function will combine. So, it will collect the all the URLs, that means group by key and it will do the summation, of how many times that number of ones are there it has to, just do a count. So, it is just like a word count program, an extension of a word count, which will find out, the URL access frequency that means our URL how many times it is being referred in a particular log file? Now, another example, another application, where it is used this MapReduce program? Is called, ;Reverse Web Link Graph’, for example there are the web pages

and web pages are, pointing to each other. Let us say this is a B and C. We want to find out that, for let us say web page see, how many different pages are pointing to it? We are given these, kind of pairs that is a is pointing to C, web page is pointing to web page C and web page be pointing to A and B is pointing to C, out of this we have to, now find out that for a particular web page, how many links are pointing to it? It's called, 'Reverse Web Link Graph'. So, it's called a, 'Reverse Web Link Graph'. So, the map function here, given this as the input, to the map function it will output the target and the source pair. So, for example here the target is C. So, C and the source. So, given AC, in the map it will emit, C and A, for each link in this, particular case similarly for BC, it will emit, C and B similarly for BA, it will emit, A and B, for each link to the target URL found in the, in a page named source. Now, the reduced function then concatenate, the list of all sorts URLs, associated with a given target URL and emits appear, that is called, 'Target and List Source' for example, here C is appearing ,two times and this list will become for C a comma B. So, for C this is, the list is, C comma, A B. So, this is, one and another one is a and B. So, this will be given to the reduced function and reduce will take this, target C and find out this list or the source. So, C, page is being pointed to by A and B, that is being computed here by, the MapReduce and the web page A, is being pointed to by only one, web link that is called A B. So, this way, popularity of the pages can also be calculated if you want to do a sum, you have to just make a count of it. So, it becomes, 2 this becomes 1 and this kind of statistics or this kind of output can be used in computing, the Page Rank.

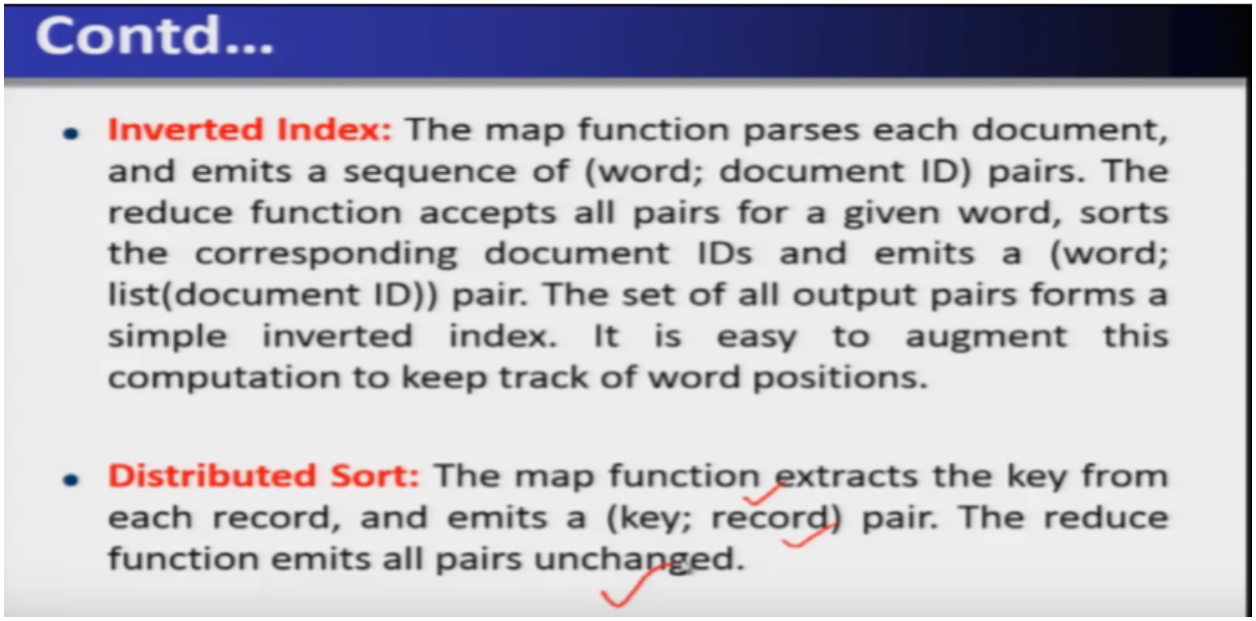
Refer Slide Time :(6: 31)



Now, another application is called, 'Term Vector', per host. So, term vector summarizes the most pardon words that occur in a document or a set of documents as a list of a word and frequency pair. So, the map function omits the hostname and the term vector, appear for each input document, where the hostname is

extracted from URL of that document, the reduced function is passed, on the all per document on vector, for a given host it at these term vector together, throwing away the infrequent terms and then omits the final hostname and term vector pair. So, that means, for a given term document we are going to find out the most important or most frequent words and we have to basically omit the hostname and the term vector pair in this particular application.

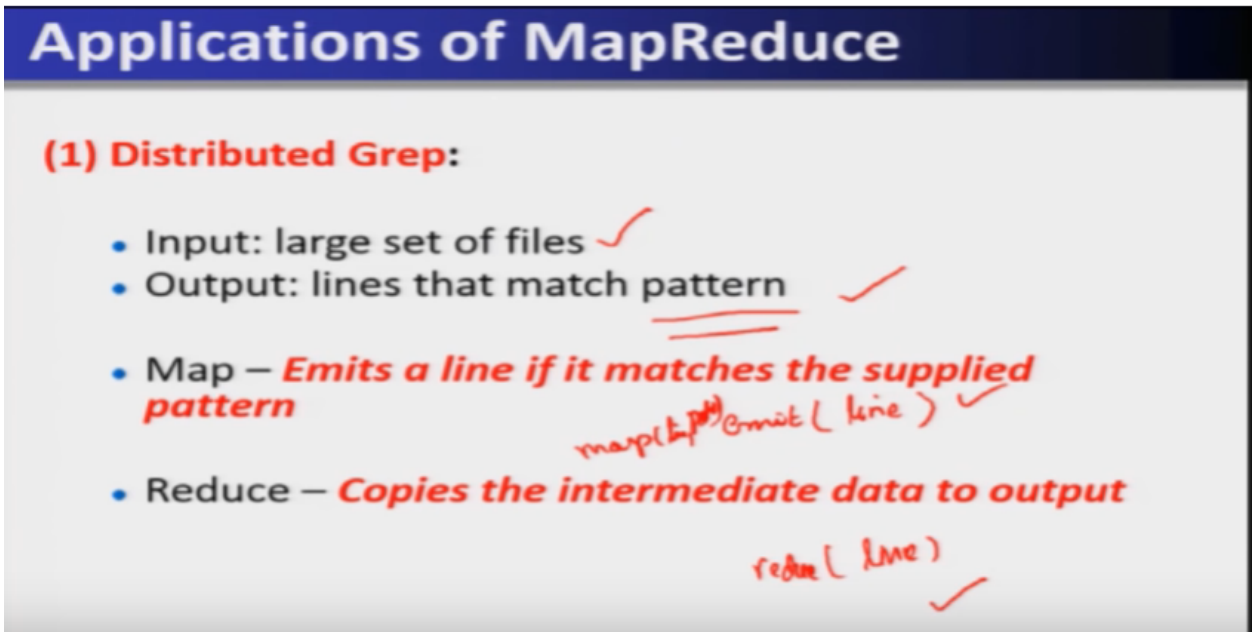
Refer Slide Time :(7:30)



Another application is about, the inverted index, which all the search engines are mostly does this, let us see, what this application? Is and how the Map Reduce can be used to program, doing this inverted index? So, here, in this application the map function parses, each document and emits a sequence of world and document ID pair, the reduced function accepts, all the pairs of a given word sorts the corresponding document IDs and emits, the word and list of document ID pair. So, the set of all output pairs forms a simple inverted index it is, easy to augment, this configuration to keep track of the word positions. So, we will, in the later slides you will see, more detail of MapReduce program for this application that is, for inverted index that is, it is, possible that if, the set of documents are given, for example search engine does this, Google when we type a keyword it gives you all the list of web pages? Where those documents? Where these keywords are appearing and that is being performed using inverted index? So, every search engine ,often computes this inverted index and that if, the number of documents is huge, the search engine like Google, are being by Microsoft all, they are, they are basically computing the inverted index and whenever the user searches it, it will perform it will check this, inverted index and gives, that particular outcome. We will see, in more details how, this MapReduce exactly program this, inverted index application, similarly the distributed sort, that is the map function extracts the key, from each record and emits, the key and record pair the reduced function just emits all the pairs unchanged ,that means automatically, internally the map function gives, when it emits the key and required, pair it provides in this sorted order and if there is, nothing different happens, into the shuffle phase, then if the

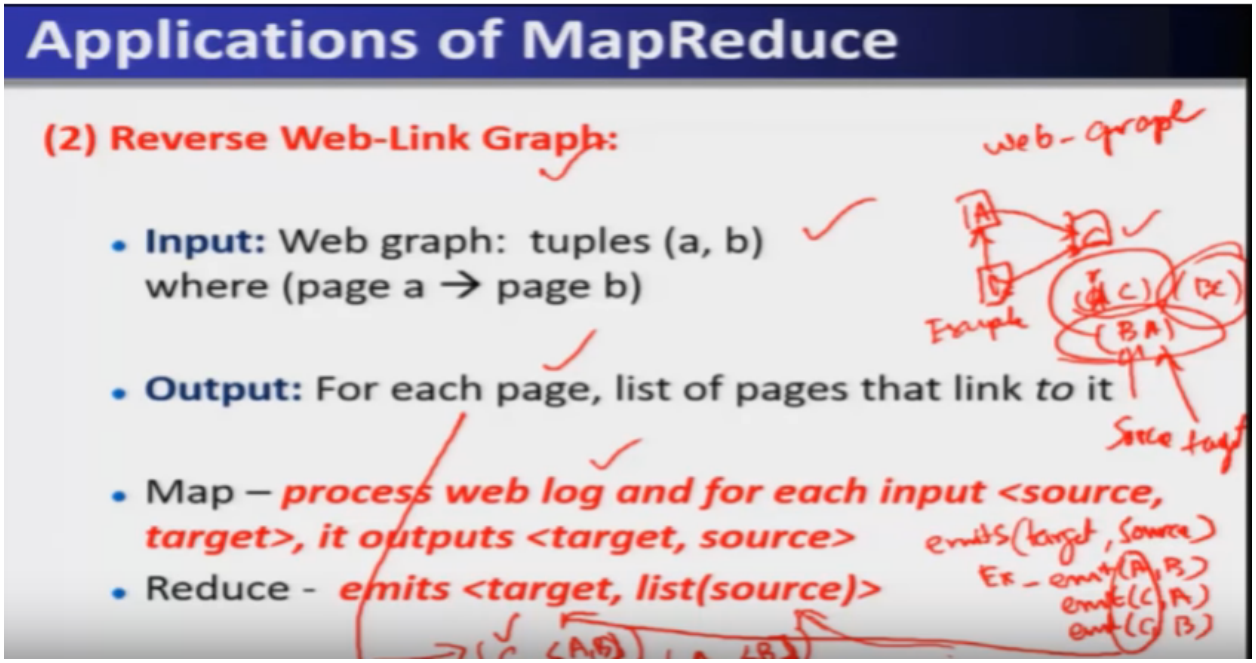
output is, given as it is, then this, particular outcome of the map face will be taken up, as and it will be emitted unchanged, by the reduced function and this will perform the distributed sort.

Refer Slide Time :(10: 18)



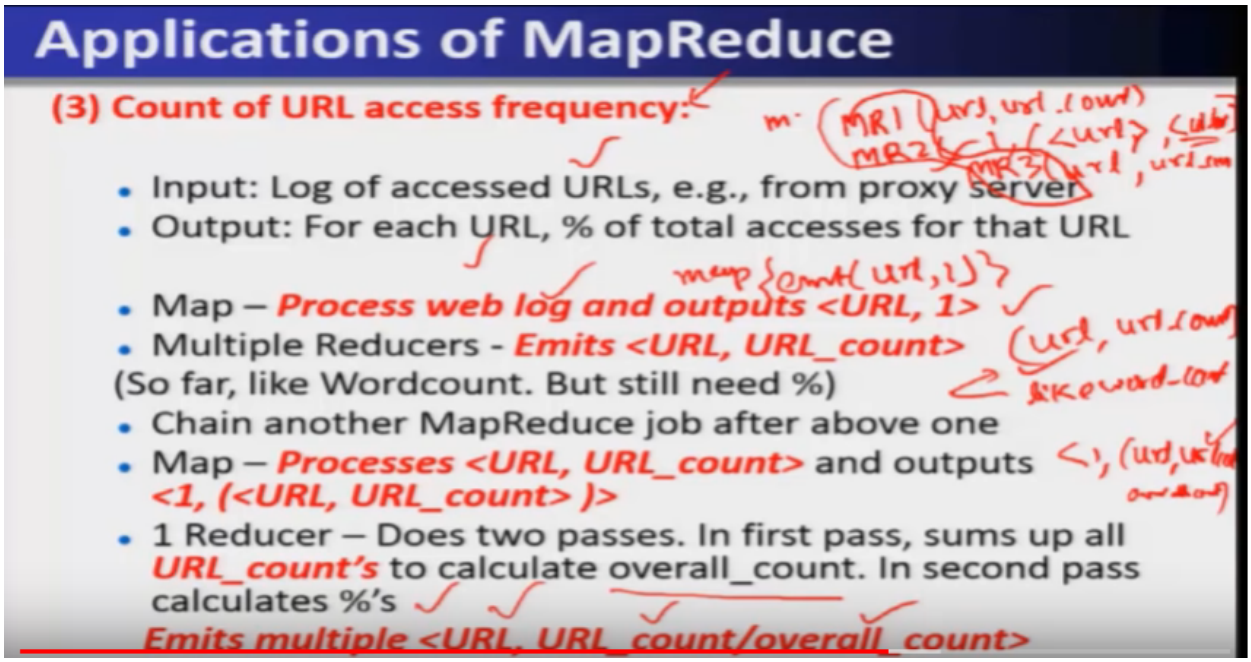
We will see, in more detail of distributed sort application, how it is done in the Map Reduce? Applications of MapReduce, we are going in now, little more details, of these applications which we have, some of them which we have summarized .Let us take the example of distributed grab how using MapReduce we can perform this distributed group operation? We assume, that the input is a large collection of files and the output P I want to get is, the lines of a file, that matches that particular pattern, that matches a given pattern. So, the map function, will emit, a line if it, matches the supplied pattern. So, map function will, emit a line. So, here the things are quite simple, why because? Whenever the line is matched, in the map function, line and a pattern. So, it emits only the line and the reducer doesn't have to do anything it will copy the, all the intermediate data, which is given by the, map function and it, will output.

Refer Slide Time :(11: 53)



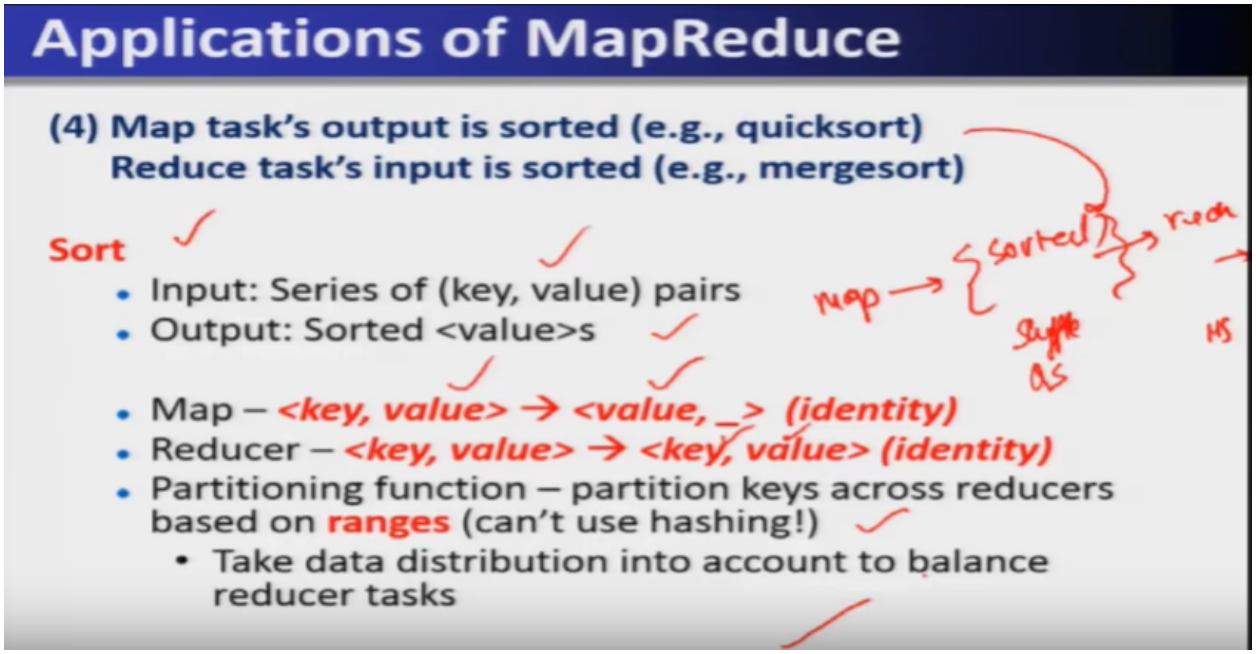
So, this will become the distributed graph. Reverse web link graph we have already, seen let us see again, for the sake of completeness and we assume, that the web graph is, given in the form of a tuples, A, B. So, again I am drawing the same picture, let us say it is, A B and C these are the web pages, web pages and let us, say that a is pointing to, C and peace' pointing to A. So, in this example, the tuples which is given as the input are AC, then BC, then BA. And so, as far as, the map is concerned map function on getting this, as the input. So, that means these are the edges, which are given as the input to the map function and that means the source and the target, source this is, the source and this is, the target. What it does is it emits the target and the source? That means, for example for BA, it will, emit a and then B and for AC it will, emit C and then A and for B C it will emit, C and then B. Now, after doing this emit, the reduced function will accept this target, target means these, things will be accepted here, in this reduced function. And we will, form the list that is the, the list of sources. So, far C, it will emit C and the list as list as, A and B this will be the output and for, for A it will emit, the B itself. So, for a page C A and B they are pointing you can see, in this particular picture, A for C A and B they are pointing to it and for, for A B is pointing to it and this is called, ‘Reverse Web Link’. So, output for each page, the list of pages that link to it that we have already achieved, in this particular application using MapReduce program. So, you can so, the programmers can easily, write down the MapReduce program for different this application we have seen, the reverse web link graph.

Refer Slide Time :(15: 14)



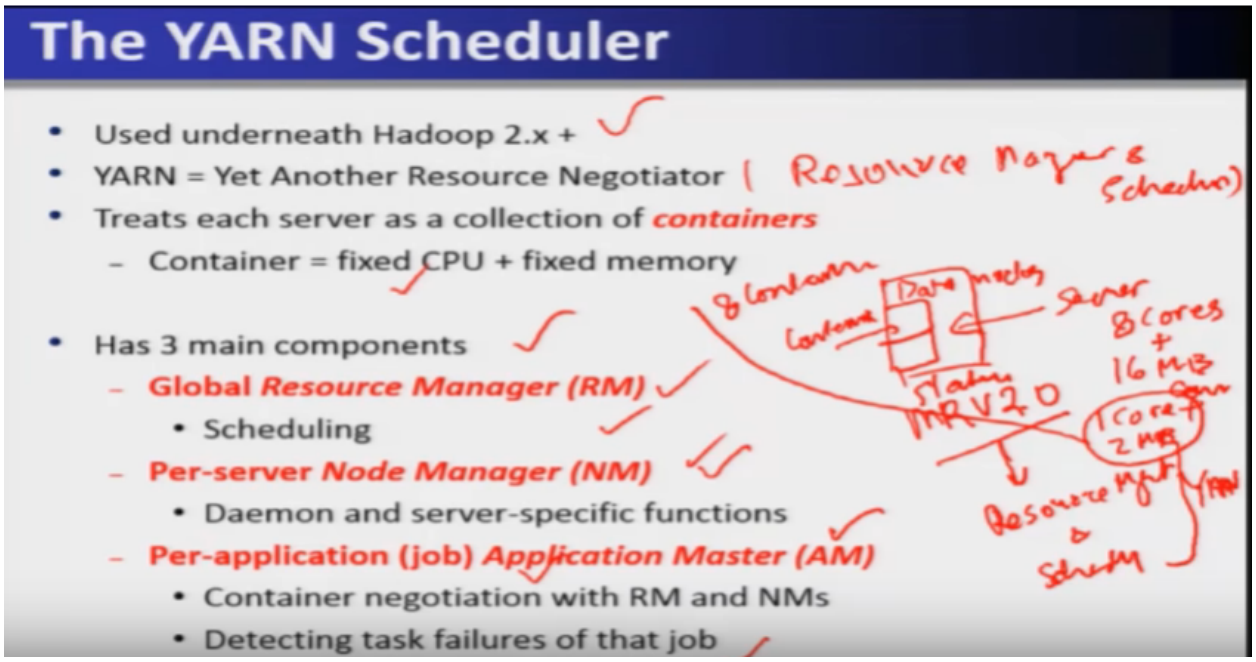
Similarly if you want to find out, if you want to count, the URL access frequency, that means the input is in the form of the log file which has accessed URLs and normally, this particular log file we can obtain from the proxy server, which maintains the log of URLs, which are accessed by the different clients. So, out of this, particular log analysis and the output we want is, that for each URL, we want to find out the percentage of total accesses, for the URLs .So, we want to find out and then rank it, later on we have some other applications. So, how to find out for a particular URL how? What is the percentage of accesses for that URL according to that log accesses? So, the map for this particular program will require to get the weblog and output and it will emit the URL, with the value 1. So, for every URL it encounters in, in the weblog, it will emit URL and with a value 1, the map function will do this, output. Now, then to find out this, access frequency, in the percentage it requires multiple, reducers. So, the first reducer, it will emit the URL and the URL count. So, that means, it will, do this, for every URL, it will also, do a count. So, it is just like, what count program? Like what count it will do a URL count? So, out of this ,particular URL count, the map function, another map function will, will execute, it will take the URL and URL count and output as 1 comma URL and its, URL count. So, after out after this, output the reducer will now, perform two different passes, in the first pass it will, sum all the URL counts, to calculate the overall count and in the second pass it will, calculate the percentage of that URL and percentage of that URL. So, it will emit the multiple URL values and URL count divided by the overall count. So, in this particular, example we have seen, not only one Map Reduce but, a series of Map Reduce. So, this is, the first Map Reduce function, will emit this, one URL and URL count and which will be taken by another Map Reduce function, which will compute, which will now, compute, which will emit, one and URL and URL count which is it will emit one and whatever values we are getting and then there is, a third Map Reduce which it will now, calculate all the sum and find out the URL. So, it will, emit the URL and URL count, divided by overall count. So, they see that, this is a chain of, MapReduce functions, which are required to solve this particular problem. So, earlier examples which we have shown only one MapReduce but, now we have shown that several sequence of MapReduce, are required to solve. So, if there is, such complicated or complex applications are there. So, it is, possible to make a chain, of MapReduce job and solve this particular problem.

Refer Slide Time :(19: 44)



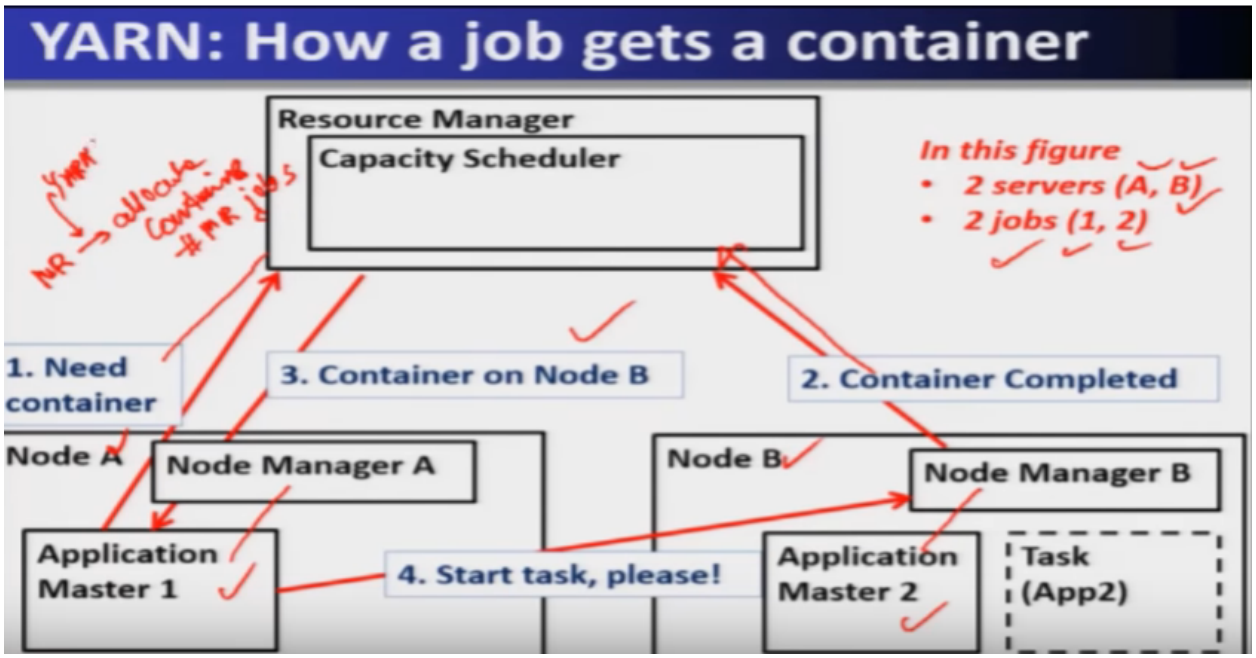
Another application we will see, about the sorting and here, the input is given in the form of a key value pairs and we want the sorted, values to be output. This particular program as we have, shown you quite simple for example, the input whatever is given to the map function key value, it will output only the value? And the reducer job, also will just output this key and value, whatever is there? So, in this particular process, when the map, outputs these values, which are already in the sorted form, normally quick sort is done, here when during the shuffle phase and if the same thing is output, in the in the reduced function. So, it passes on and it uses the mud shot. So, it is a quick sort and the reduced function uses the merge sort. So, quick sort and merge sort together, will sort the applications and we don't have to do much, the partitioning function we have to be careful ,during the sort is that partitioning, partition keys across the reducer, is based on the on the on the ranges and you cannot use hashing, otherwise it will disturb the sorted order. Okay?

Refer Slide Time :(21: 14)



The YARN scheduler. So, for MapReduce, job scheduling and resource management, YARN is used. So, let us see and go through the YARN scheduler in more detail because, in Map Reduce version 2, 2.0 the scheduling resource management and she dueling, is done by the YARN. So, let us see, how the YARN does this she dueling? So, it is used underneath, the Hadoop 2.0 versions and onwards. So, YARN full form is yet, another resource negotiator and its job is the, the resource manager and scheduler. Now, this YARN treats each server as a, collection of containers. So, by this means is that so, the data nodes which are called as, 'Slaves'. Is in the form of containers, containers is, a container is having a CPU and a fixed memory. So, for example if let us say, a data node or this machine or a server ,has let us say, eight cores and has some memory, let us say 16 MB of space. So, the container will contain, a one core and let us say, 2 MB of space this is, one container. So, it will how, an four eight different container in this example, eight containers, in this, configuration. So, it depends upon, how many cores are there in the server? So, that many number of containers and the memory together can form the containers and container is the unit, which is being allocated, by the YARN scheduler, for MapReduce job. So, it has three different components, YARN has three different components, one is called, 'Global Resource Manager' and which performs the overall scheduling and for per, node manager it is called, 'Per Node Manager'. It's called, 'Node Manager' also, daemon and the server functions, it will specify and for per application, that is the job ,there is another job ,application master and this application master will negotiate, with YARN, for getting the container, with there is, the resource manager and the node manager and whenever it detects a for failures? For that particular job the application master again contacts the, the yarn component that is the resource manager and a node manager.

Refer Slide Time :(24: 22)



Let us see, how this all flows are in this YARN between the resource manager and node manager? So, let us say that, we have two servers A and B, A and B there are two servers, A and B and we have a two jobs, one and two which are to be allocated. So, how it does is first of all, this particular resource manager is contacted and it knows, about the scheduling or it will schedule these jobs to be allocated? The containers and then it will be shade ruled over there. So, this will, this client will, give a request to the to the resource manager, that is a component of a YARN, for she ruling these two jobs and this particular resource manager knows, that there are two servers A and B, with the available container, within it. So, let us assume, that there is one container available, on B and there is, containers available on A also. So, these containers, after allocate after allocation, it is informed to the application master and the application master in turn, contact with the node manager and starts, the execution on these containers and once the application, execution is over, then these containers, then this application master will inform and the containers will be returned back. So, just see that, that means MapReduce jobs, with the help of YARN, MapReduce jobs, with the help of YARN, allocates the container, for that many number of, for the number of MapReduce jobs. So, this is done, through the help of YARN. So, this is explained here, in this particular picture. Thank you.