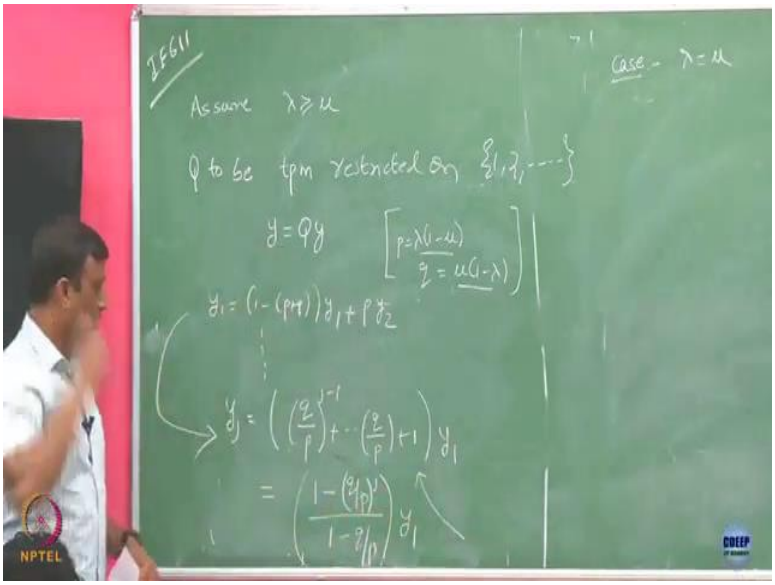
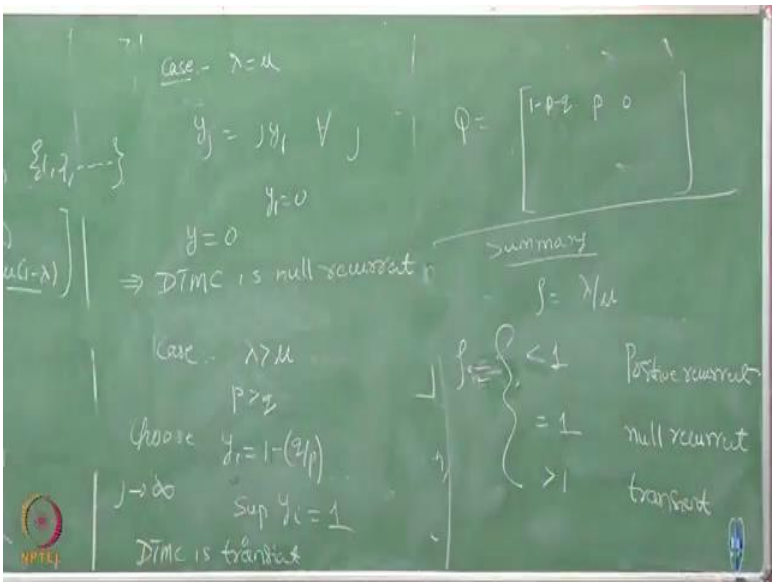
Introduction to Stochastic Processes
Professor Manjesh Hanawal
Industrial Engineering and Operations Research
Indian Institute of Technology, Bombay
Lecture 42
Queue continued and
Example of a Page Rank

So, what conditions we know to verify it is a positive, it is a transient or null-recurrent. So, now that under lambda is going to be greater than or equals to Mu I know that my Markov chains cannot be positive recurrent. It has to be either transient or null-recurrent. But I know a condition, if I somehow know that my DTMC is transient that is enough for me. If I know it is transient, if it cannot be transient, it is null recurrent. If it is transient, it cannot be null recurrent, right.

So, let us try to verify my Markov chain is going to be transient or even recurrent. I know if it is a recurrent when lambda is going to be greater than or equals to Mu, it has to be necessarily null recurrent, it can be positive recurrent. So, what is the result I know in that case?

(Refer Slide Time: 01:20)





So, we already know that if I have a reduce transition probability matrix Queue. And if I can if I have a y equals to Q y solution, is that y happens to be 0 vector then I know it is going to be recurrent. So, maybe let us try to apply that.

So, in this case, if I just know if it is recurrent it is enough right, if it is no recurrent, it has to a null recurrent. And if it is not recurrent, I already know it is to be transient. So, now consider Q to be tpm restricted on 1, 2 like this. So, I have just excluded state 0 from my state space.

And now I have my Q to be restricted to these sets here. And now let us try to look at the solution of Q is equal to Qy. And if my y equals to Qy, solution happens to be here 0. Then I know it is going to be recurrent, right. So, let us verify whether that is true. So, in this radio

transition probability matrix, so I guess, just for notation, I am going to write this P to denote lambda into 1 minus Mu and Q to denote Mu into 1 minus lambda, because these terms come to us many times, so just write them as P and Q.

Now, if I want to write the solution for this y equals to Q y. What I am going to do, what I will get is simply, I am just going to write a simplified version you can verify this. And like after doing some manipulation over this like the way we did when we try to solve Pi equals to pie P, what you will end up is y j that is the jth term here can be written as q by P to the power j minus 1 plus q by P plus 1 into y 1.

So I am just skipping the process like if you just write this iteratively, okay check this anyway. What will end up is finally, this iterative equations like way the jth component, I could write in terms of y 1. Okay so, if you further simplify this, it is Q y2, so let us quickly check that, why is that? So, how is my Q is going to look like? What is my first row is going to be?

1 minus P minus q, so what is this second row is going to say? It is going to say, from 1, I go back to 0, right? So, I have removed s, it is going 1 to 1. So, this is 1 minus P minus q. And what is the, so I am interested in a row here right, yes. So, what is the second component? This is going from 1 to state 2. It is just going to be P and after this it is going to be 0 here, right.

So, what I will get? And now, when I multiply it with a row, column vector y 1, y 2 here, I am going to get this into y 1 into P y 2, right, so that should be correct. And now if you apply the geometry, so again, now I help q by P here. So, if I am going to treat it as a geometric progression here, with the ratio time q by P, what is that you are going to get here? 1 minus q by P to the power j divided by 1 minus q by P into y 1, right.

I have this, I have in the case where lambda is going to be greater than or equals to mu. So, further assume now, consider the case, only where lambda is equals to mu itself. Then I will separately deal with the case when lambda is going to be greater than mu. So, when lambda is equals to Mu, what happens? When lambda Mu are equal, p and q are going to be equal, right?

Sorry P and Q are going to be equal, what is this quantity is going to be? Again this is 0, but in the, we can just look into this case here before we, it is going to be, if P equals to q equals

to P right, this is simply going to be, this sum is going to be j, j y. So, again, what I am going to get, y j is equals to j y 1, and this is true for all j okay.

Now, here y 1 is going to be free variable and y j depends on that y 1. Now let us say, you select some y 1 be something between 0 and 1. And now, my j is all possible values right, it can be 1, 2, up to. So, then in that case there exists some j large enough which will make this y j to be greater than 1, right. Whatever y, you know, whatever y 1 we are going to choose some positive value, it is going to make y j to be greater than 1 at some point for some j.

So, because of that, you will not it will end up with the y which is between 0, 1 right. So, I am seeking a solution of this which is in this interval not any y. So, if I start with any y 1 positive value, I will end up with y j which are going to be larger than 1 and then in that case my y vector cannot be in this region, is that clear?

So, because of this, if I want to restrict my all y js to be within the interval 0, 1 any positive value of y 1 will not do. The only possible value y 1, I can choose is 0. But if I chose y 1 to be 0, all my y js are going to be 0. So, then y equals to 0 is my solution. So, if y equals to 0 is my solution than what I know? It is recurrent, but I already know that it cannot be positive recurrent. So it has to be null-recurrent.

Now, what happens with for the case where lambda is strictly greater than Mu? What is that? So, when lambda is going to be strictly greater than mu, so P is going to be greater than q right. So, if P is going to be greater than q, this ratio here is going to be less than 1. And you will see that if you choose your y 1 to be simply 1 minus q by P, all these guys are going to be less than 1, a solution of this.

So, if you just start with this y 1 equals to 1 minus q by P which is and then plug back here you get y j. So, notice that here y j is need not sum to 1. All we need is this y j is to be between 0 and 1. So, if you start with this y 1 like this, you will end up with all these y js chairs which are anyway positive, but you will also notice that there are strictly less than or equals to 1.

But as j tends to infinity, you can see that, that j tends to infinity this guy is going to get 0, but 1 minus q P is same as y 1 and this y j tends to 1. And we have, this is also consistent with our earlier observation, right. The solution will be such that the solution y will be such that either it is going to be all zero or it will be such that supremum of y i for all i is going to be 1

here. So, now the solution is such that as j tends to infinity, this is going to be y 1. Okay in summary.

Student: (())(12:03)

Professor: Yes, that is what we say in this case, we have this y vector which is not 0 like this, it is something positive here. Now, we know that if it the solution to this is not 0, then it is transient. So, just summarize, so let us write my row to be lambda by Mu. So, what we showed? When rho is strictly less than 1, my DTMC is what? So, when this is, we showed that it is positive recurrent and when it is equals to, sorry strictly less than 1. When it is equal to 1 and it is greater than 1 that is going to be.

Okay now coming back to our queue example we had, we had a queue where customers are joining and they are getting served. Lambda is like arrival rate and mu is like my service rate, right. So, we can think of this ratio lambda by Mu as kind of load factor, how much load I am putting into the system right. When this load is less than 1, what I expect, what I expect like, so when this load is less than 1 what I, what is happening is, service is happening at a faster rate compared to the arrival rate right.

So, that means I can flush out the customers faster than they arrive okay. So, because of this, I expect my queue to be positive recurrent here because I am flushing out them, I can expect my state particular state to be getting visited again and again, what is my state? The number of customers in the queue, I can be revisiting those states again and again frequently. Okay so before this, let us focus on the case where lambda is going to be strictly greater than one.

When lambda is going to be strictly greater than one, what does this imply? You are here doing service at a smaller rate compared to the arrival rate, right that means you are basically your system is slow. So, when your system is slow, what you expect? So we expect your queue to blow up, right. So, that is why transient. So, what is transient here imply? So, transient is implying you start from any state, you do not come back to that state.

So, you start your system in any finite state that means, you will not come back to that finite state again that means you are going to explode. You start from any finite state, the probability of coming back to that state you there is a positive probability that you will go out of that finite state right. And this is true for any finite state because the whole DTMC is transient right. So, you start from any state, finite state there is a positive probability that you do not return to that state.

That means not return to that state means we are basically exploiting. And when n equals to your arrival rate is just equal to a service rate. That means, you may be able to flush out, but that flushing is happening very, not very often it is happening very rarely right. So, there is a possibility that your queue can build up at some point. So, if you are a designer or some queuing system, or any customer or some processing systems, so where you have to process, some requirements.

It could be like somebody asking for a ticket or some computer systems, asking for some service or whatever jobs that are you have to deal with. So, this is basically saying that I would like to design my system such a way that my service rate has to be larger than my (service) my arrival rate.

So, arrival rate you can think of job arrivals or whatever, jobs could be I mean computer request or buying tickets or like whatever somebody asking for some particular service. So, this is what kind of also tell us when my system is going to be stable. If you define your stability to be that my queue never blows up, my queue will be always take some finite states and it will never blow up. Then if you want to stabilize your queue or your, you want to stabilize your process, your operation. You want to ensure that, you want to be better be in this regime, positive recurrent, right.

So, even when you just let lambda equals to mu, it is possible that you may not always be in a good state. But if you let lambda rho to be less than 1, you are going to, you will keep coming to a finite state. Finite state could be one of the state could be 0 state also right. You will start with a 0 state that means you have side everybody you will keep coming back to that state again and again and that is with high frequency.

So, if you do this, you may still come back to that but that may very happen very infrequently, okay so fine. So, this is one example of where a priori knowing, what kind of states my DTMC takes, either transient recurrent, a null recurrent. I know whether my system is going to be stable or not. Okay and accordingly, if you know some arrival rate, you want to set up your service rate such that that exceeds your arrival rate.

Okay now, so let us look at the other examples of where my knowledge of details is going to come to our head. So, how many of you know this page rank algorithm? Yeah, only one, so all of you do search right, all of you is a search engines. Right now all of us only know

Google search engine, because Google is so popular that it has killed all other search engines in the market.

So before that, there were quite a few and what is the algorithm that works behind the search engines? So, one of the ideas is also derived from this in what we had this invariant probability distributions Pi equal to pi P. So, let us try to understand how is that? So, in internet, you have so many pages right, HTML pages and you are looking for some content, it will be available at some HTML page.

So, if you give that, what do you expect, what you want actually? You want, you to Google or any search engine to list give the link that has the most relevant information you are searching for, right. Okays so, may be if you just want to talk about IIT Bombay, you want Google to first show IIT Bombay homepage, rather than some blog talking about IIT Bombay right.

So, Google has to somehow rank all these pages, so that it gives you the most relevant information. So, how to do that? Okay so, obviously, you want to kind of do ranking of these pages based on that you want to list, now how to do the ranking. So, can anybody think of some simple way of ranking pages?

So, one possibility to rank them based on the, what they call it as let me see the term, what that call as, how many other HTML pages that links to this page for this information? So, if other pages are talking about IIT Bombay, how many are them referencing to this page?

So the page which is most cited, you want to bring them right, suppose let us say, you are searching for some research paper or some you want to understand some topic, let us say for time being DTMC, when you search on DTMC, let say there is one paper which has been heavily cited by many researchers. And that you like to read or something who have just used one DTMC word somewhere you want to use it.

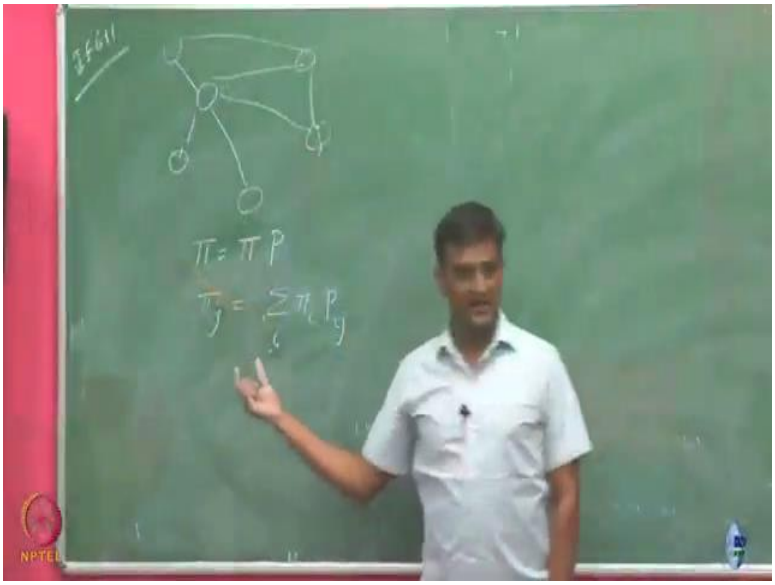
So, you want to definitely use for once which is most cited. So, because that has been more popular and possibly that is the most informative about a DTMCs. So, in that way that the once, which have high citations can be preferred right, when you want to rank. So the one guy with the highest citation can come first, then the next one like that in that way you can order.

Now, the question is how to do this or do the citation count itself. For example, as I said, I created one page and I myself created some thousand other page beside my work. So, let us say, I want to increase my citation and I write some crappy paper and write another thousand crappy paper which talks about this crappy paper. So then, Google is just counting right, how many guys are referring to this.

So, that will automatically boost up this crappy paper up. So, I also do not want this, so what I want is, the guys who are more citations, who are referring to this should be isolated more right. So, suppose some thousand crappy papers are referring to this, I do not want to give equal weightage to all of them.

So, maybe among them maybe there are some authentic papers also good ones, which are referring to this, maybe they want to be given higher preference. So, how can we do this? So, when I think these people were all thinking about how to do such kind of ranking, they possibly realized that I mean, they realized that this is nothing but a solution to Pi equals to Pi P, why is that?

(Refer Slide Time: 24:56)



Suppose let us say I have this several loads, think of each one of them some HTML pages, they have a cross linking like they refer to each whatever manner. Just think of some referencing there happening. And each one of them have some importance okay. Suppose I am going to think of, I am going to think each one of them as one particular state, each HTML page as a state and then Pi to be the some connections basically.

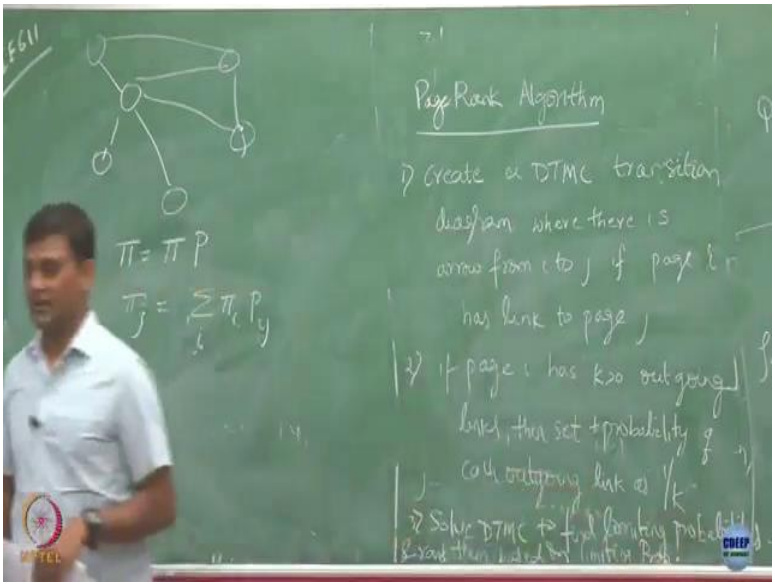
So, if there is a connection between one link to another, based on that I can define the appropriate probabilities, we will just mention how it is. So, then we can think about that as Pi equals to Pi P where I am going to start here. What is this Pi? This is Pi I of P. This is ij or ji? So, this is going to be ij then this is going to be j right. So, Pi j equals to summation Pi I, Pi ij.

So, here it is telling how many pages that are linking to me right. So, I am interested in a particular j let us say, now this is telling how many of them are connecting to me that is captured by this P and then P i is going to give me the associated weight. So, my weight Pi j going to be higher if the P i js that are connect to me also higher, right. So, my preference is going to be higher if the other preference guide are also connecting to me.

Thus this capture this equation capture that, so suppose let say I have given weights, some preference to all the pages, in the net and they are connecting to me. If the preferences are higher and they are connecting to me, my preferences will also be boosted right. If a high preference guy connects to me, my preference will also get boosted right.

So, in that way, you can try to capture the importance of a page, you have in the internet and based on those importance that is now captured by this pi j. You can order the pages and then display to you. So, let us, just in summary, now how to come up with this P ij is the question right. Okay so, let me write algorithm for this.

(Refer Slide Time: 28:15)



So, this is the one crude version of algorithm. So, what you do? You take all the possible pages and draw a link between page i to page j. If page i is referring to page j okay and then

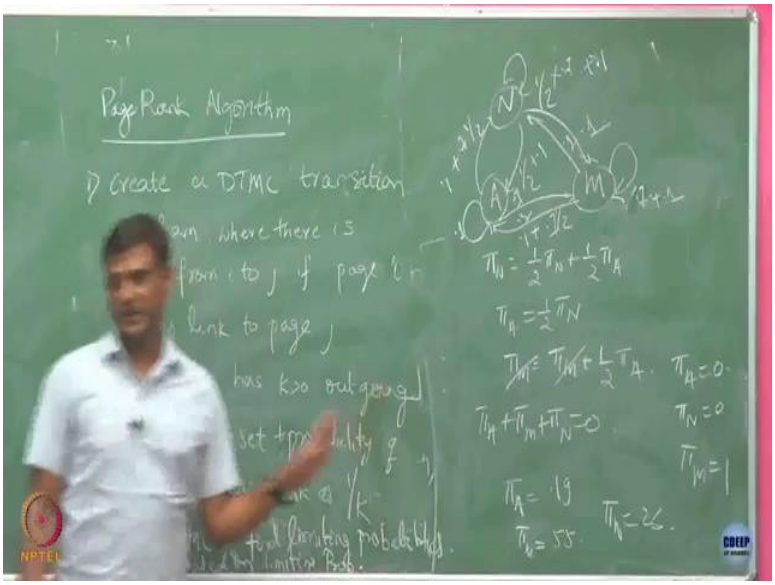
take a particular page and see that how many pages it is referring to? Suppose if it is referring to K pages, you it has a K outgoing links and give a weight of 1 by K on each of these links okay.

So, that is it, now you are not giving any importance there on each, you are just treating all the outgoing link equally. And now once you have this, you have a P matrix and then solve Pi equals to pi j. And on that basis, you are going to rank your pages based on the values of Pi you are going to get okay.

So, if you have a web page, you can just try to see, what is the page rank of your web page, I do not know if Google displays it for all pages, but if you are interested just see, what is the page rank of yours and you can just also find out which page has the highest page rank that tells which is the most visited web page.

Okay so, this is a very crude level thing, the page rank, but it has it has not that if we just simply do like this, it is going to give you the best solution. So, just to motivate what could be the issues. So just for us, for simplicity, just look into this one example.

(Refer Slide Time: 32:40)



So, let us say there are 3 states like this N, M and these are the transitions, so I am just drawing 3 pages and let us say I have been assigned weights in this fashion. So, if you solve this DTMC for its limiting distribution, what you are going to get is simply, so I am just quickly writing this the equation you are going to get is.

So Pi of s going to be half Pi of 1 and that is only incoming link there and then Pi of M is going to be Pi of M plus half Pi of A. So, if you just solve this, what you are going to get, and now you anyway, you also have this constraint. So, if you just solve this, you are going to get Pi A equals to 0 and once Pi A equals to 0, then you are going to get also Pi N equals to 0 and then you are going to get Pi M equals to 0, Pi M is equal to 1 this in this case, right.

So, what is happening in this? If you just blindly take this invariant probability metric, what you are basically doing is, you are a basically and trying to give most preference to this debt trap here. So, I am going to call this is a trap here, because once you hit this link, it is always self-flow. It is redirecting to itself, it has a self-flow.

So, because of this, you will end up with giving preference to this the most, whereas ideally, you do not want it possibly give a preference to this because this guy has at least link from, it has more connection than this right. So, if you are just going to do like this, we will not end up with a good probability distribution.

So, there are to improve this, there are other methods that are some are (())(35:19) and some are I think, well developed. So, one possible think is when you have things like this, you try to redistribute the weights, one possibility is whatever the, so right now this guy is not referring to anybody. You forcefully when you get trapped here, you try to come out of this, how you are going to come out of this?

You try to deliberately add an outgoing link to this and assign some probability to this. So, one possibility is what they call it as, taxing it. So what you do is, take out some percent of the tax, that is some probabilities from each of this link and distribute that tax to all among all the states.

So, one possibility let us say, I am going to tax each of these links by 70 percent, okay what I mean by that is, all these probabilities I am going to reduce to 70 percent. Okay so, in that case, this probability is going to be 0.7, everything is going to be 0.7. This is going to be 0.7, this is also going to be 0.7 and this is going to be 0.7.

And now I have said about 30 percent from each of these links. That I am going to distribute among the states. So now, so this, so I have 3 possible things right from here to here and also self-link. So, the self, I am going to add 0.1, this part, I am going to add 0.1 and I am going to add a link here 0.1. So, whatever the 70 percent, I had 30 percent I have taken out, now I have just redistributed. And I will do the same thing.

Student: (())(37:21)

Professor: Yeah so, I have just taken out 30 percent from here right and I have just redistributed. So, it will nothing have changed. And now if you again, so I have taken 30 percent out of this right, so I add 0.1 here. And this one also, I add 0.1 and also add a self-loop here with 0.1. And similarly this guy also initially did not have any link to this and also this, so I will add 0.1 here 0.1 here and the remaining 0.1 here.

Now, if you do this kind of perturbation through this taxation on this, now we will end up with another set of transition probabilities here. And on that we will end up with a new set of values, which I am just writing here which is at least better than what we had previously. So, now it has kind of distributed the weights.

But still, as you can see that this new probabilities will depend on how this, how much is the tax that has been put? If you are going to change this tax, you are going to get a different one. And the that thing is again, if you are going to tax heavy, load the tax large then so suppose you are going to tax 100 percent right that means you are basically equally distributing the links, all of them.

And you are going to make the tax 0 very low, then again you are going to be coming back to a solution which is close to this. So, one has to appropriately choose this taxes, so that you get the correct ranking. And there are many input questions based on this. So, we will just leave it here. So basically, what we see is that in some way by looking at this equation, pi equals to pi P, if I am going to choose my pi matrix appropriately, so this will the solution of this will yield somehow the preference which captures the preference of others.

And in that way, this is going to be a good ranking okay. So, let us stop here and with this, we will just conclude this detail say discussion. In the next two classes, I want to just talk about a little bit renewal theory. So, renewal theory is kind of, you see that some of the concepts we have already discussed they are just a more generalized questions of that.