Parallel Computing
Prof. Subodh Kumar
Department of Computer Science & Engineering
Indian Institute of Technology – Delhi

Module No # 03
Lecture No # 11
Memory Consistency

Let us begin we were taking about the NC class and although we will discuss about binary I was going to bring it up we are going to talk about parallel algorithm parallel program design later on also let us spend just a couple of moments I want to search in parallel forget about binary search and so although binary search is the context in the background which is what we start we start with by saying that something can search through started less log in time can we do better using a polynomial process of numbers.

And of the thing you might remember was I said that you want to make as big sequential sections as you like as you can and then try to postpone the synchronization for later right I would say great way to do this would be I take N processors I gave everybody a length one part of my array right. How will I take to search? Order one so is it parallelizable so it can be agreed it can be done in order as a matter of fact we did not do any binary search at all right you might as well have had the array completely sorted I would still be able to find it in order.

But how much work did I do order N which means now if I implemented on P processor machine I expect to be how long N by P right and that is not comparable to log N that is what we are trying to do binary search except we said we have N processors we will give everybody a sequence of length one a binary search right it is just degenerated into regular search into sequential search.

And so speed over that is good except the work done was too much the time no that is same algorithm. Algorithm is everybody checks if they have the same element and if they have it they raise their hand right so how they say is we got it so that specific algorithm which not implemented using work scheduling principle on P processor machine we will take N over P time.

Because the work done is order time we would like someone to do less than order n it does not matter how many processor you begin N over P is time taken on a P processor machine for the order one algorithm okay. And now I said everybody take one over P of P original list you got it sorted and now you take log N time log N over P time right. If that is your algorithm how long does it take overall?

That is what did we said instead of searching through the way original algorithm would have been doing now you do binary search in your share of N over P elements and that you can do N over log of N over P and one of them will found it and raise it can you do better. So yeah I missed that part how much is the work done T times log of N over P because everything is going the work in that part.

Can you do better than that actually may be the work directly does not make sense in that this context also because we have already gone to N P processors so it does not add to much to the discussion we would have got very close to log N anyway right log N is the binary search time and we have said instead of log N we are turned it into log N over P can we yeah so that something that you would consider an improvement right something that was polynomial was turned into poly log that is a good that is a parallelizable.

Similarly something is log already turned it into sub log already when you turn it into sub log another alternate might be you instead of saying you take one PF of it and research in it because very quickly you will know you do not have it right so when you do this binary search one of them as it you do not have it so know immediately that is not in your range and then you are sitting ideal right at that one guy as it is going to take all the time.

So that is not the great load balance so the algorithm now that is you take the array instead of checking the element you looking for is the right of the middle or left of the middle you divide it into N chunks P chunks P is number of processors and you decide the first processors decides the left of the N over P position or right of it the second decides if it is left is less than N over Y of that or right of it and so you have got 3 different places where this test is being done.

You are not a onetime everybody has done one time first and how do we know who knows that all is left but who is going to check that global memory then the master will take entire to find out it is not simply that am checking whether it is to my left or my right and saying is it in my range right you can do it by doing two test in order to doing one test and then checking what your left labor found right everybody checks everybody does their test and checks the next level test.

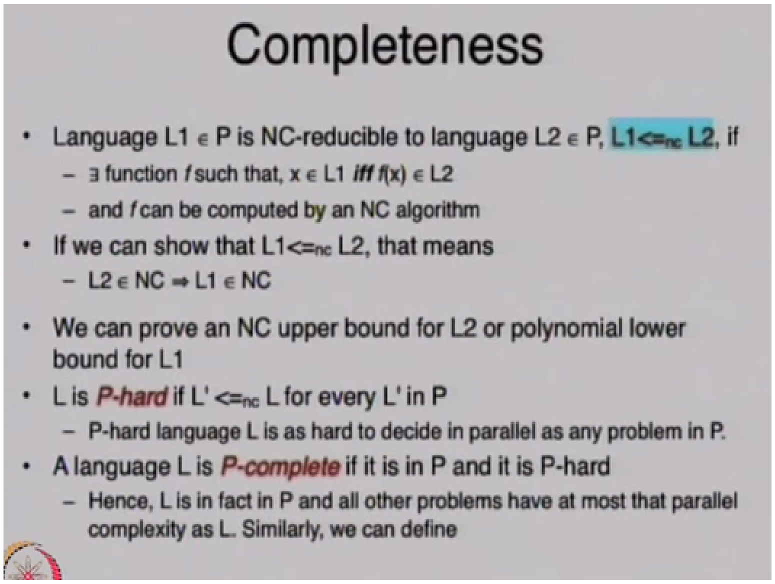
It is sequential and only one will know that their test and left neighbor test were different there are two comparison either way two is fine so in one case the comparison is whatever the search comparison is which is expected to be heavy duty in other case we are looking at the results of your neighbor but still there is two comparison right but now how time it takes now everybody has to be that guy determined it so there is one location where somebody as written left range is not 0 anymore it is this L1 right range is not N -1 anymore it is 2.

And then everybody does a sub division of L1 and L2 the gap L1 so you going from N, N over P, N over P square right so when thus you get 2, 1 log of N to base P which is same thing right log N over log P and the one original the earlier algorithm which did a lot of wasted work am I did that to log N – log P. Log N divided by P so it went down a little bit it has lot of waster work now we are not wasting any work at everybody is busy all the time you went down a factor now but still factor of a past okay.

How much work is done here again it is P processor doing over certain number of time right so P time hence log N over P over log P which is more than log N the thing is that you are not doing a syntactically correct amount of work at log N is the time spent by the serial algorithm and you cannot do less work for that. If you can do less work then the serial algorithm then you have found out faster serial algorithm right.

So the amount of work done by a serial algorithm is has to be more greater than equal to the amount done by the serial algorithm. If you doing log in amount of work then you know that asymptotically get better in terms of work. Even with that kind of work the time is at least so far high okay let us get on.

(Refer Slide Time: 13:25)



So there is the same notion of NP hardness or NP completeness except whatever was NP from the sequential domain is P now in primal domain right in the parallel domain. And so there is except notion of reduction is also slightly changed whatever was NP as turned into P and whatever was P has turned into NC and what was polynomial time reducible as turned into NC reducible in parallel in using a polynomial steps okay.

So let us look at this definition very quickly and then they will be up on slides they will be go through them also this is more so you understand the concepts and understand the kind of the landscape in terms of what are the art problem from the point of view of parallel programming and some not so harder.

So we say that language and this is now determine in formal languages written in formal languages there is also a functional variant of all of this but simply say think of it as decidable problem we will have a question Language L1 and we want to know if the language L1 or deciding if something belongs to language L1 if L1 belongs to P that means effectively you are saying that it there is a problem that can be solved in polynomial time.

That language recognition problem right now we are basically defining everything not in terms of taking everything and solving it but having an equivalent language and asking if an element belongs to this language recognizing this line. And say this L1 belongs to P is NC reducible to another language L2 although it is polynomial language right which means its recognition can be done in polynomial time.

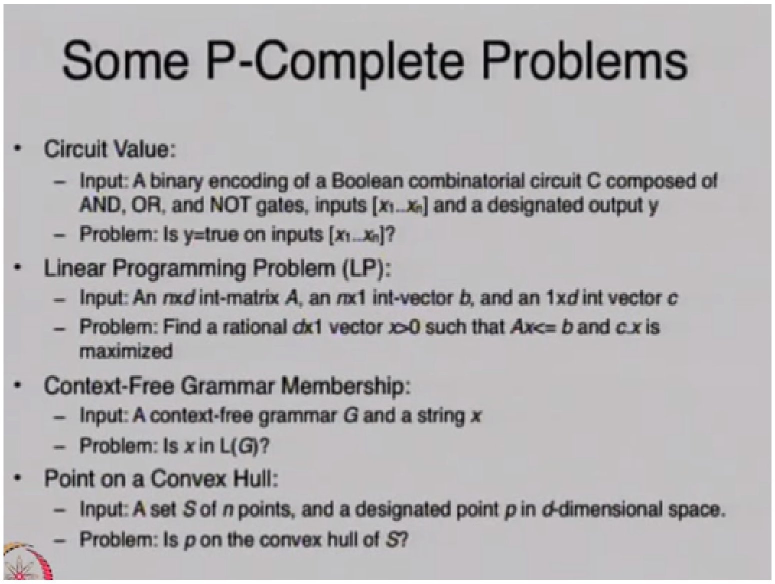
Then we use this notion that L1 is less than equal to sub and C L2 if there is such a function F that we can map from L1 to L2 right if given me a element in L1 I can apply F and get an element 2 and the computation of the F getting the element in L2 is efficient parallel efficient it is an NC. If I can compute this efficiently and I can recognize the L2 efficiently that is I can recognize in one equation.

Because I do not know how to recognize L1 I know how to recognize L2 I know how to map a problem from L1 to L2 and so I simply map the problem and call this L2 to answer that question okay. It is one way other way around something else in fact if L1 is reducible to L2 that means that L2 belonging to NC implies that L1 belongs to that means if L2 is known to be fast parallelizable then we have shown that L1 is also parallel and vice versa right.

Meaning that if L1 is not in NC L1 is known to be inherently sequential then L2 will also to be sequential is the same A implies B means if A then B as to have happen but if not B then not A also again problem L again a language L is P hard if you can take any other language L prime which can be reduced LC reduced to be L for every prime is LP whether or not efficient okay. So it says very similar is basically the same definition has MP hard reducible has become NC reducible instead of polynomial reducible and you have to replace P with NP everywhere okay.

P hard problem is something that we expect we think or not efficiently parallelizable right where MP hard as the same conversation definition of MP hard is the same and MP hard will says the language will be MP hard if you can other languages has a prime to it in polynomial time for every other language which is known to be in MP and in MP hard there is a subset that we call MP complete similarly P hard will be subset call MP the P complete where they map each other okay then can be reduce to each other.

(Refer Slide Time: 19:19)



So this is just to give you an idea that the same thing apply but this is the main point of all of that discussion is to look a couple of problem that are not know to be LC that is do not as of now have a efficient parallel algorithm we do not expect just like we expect P is not equal to MP we also expect that P is not equal to LCI. So circuit value problem is simply I have got a bunch of inputs got a bunch of gates and you tell me the output and I will tell you whether or not that set of inputs given the going through this gates produce 0 at that output or not.

And linear programming is another it is can be quite done with sequential order N linear time and so it is not even in that class of already logarithmic like binary search okay. It can be done quite efficiently in a sequential time not efficiently now from NC point of view but still you cannot reduce it to logarithmic poly log times solution and linear programming basically says that I have got bunch of constraints linear constraints in any dimension B and I want to make sure that a point is so the positive or negative half space of each of those things okay.

And that is the feasibility version feasibility problem version they are the full linear program version is the minimization and maximization of some objective function within that constraints. So this is just formulated now in that fashion compiler not easily parallelizable checking if some input belongs to a given context grammar for example and have solved efficiently in parallel and from geometry checking if some point lies on convex solve are not been done efficiently in parallel right.

Convex solve is set of the points such that their convex combination is that is the statement of course have a convex combination. And convex of set of point is the set that any line connection to point on the whole remains inside that hall inside the convex hall they all different types of instructions and then at a high level that you can say that they are inherently sequential parts to it.

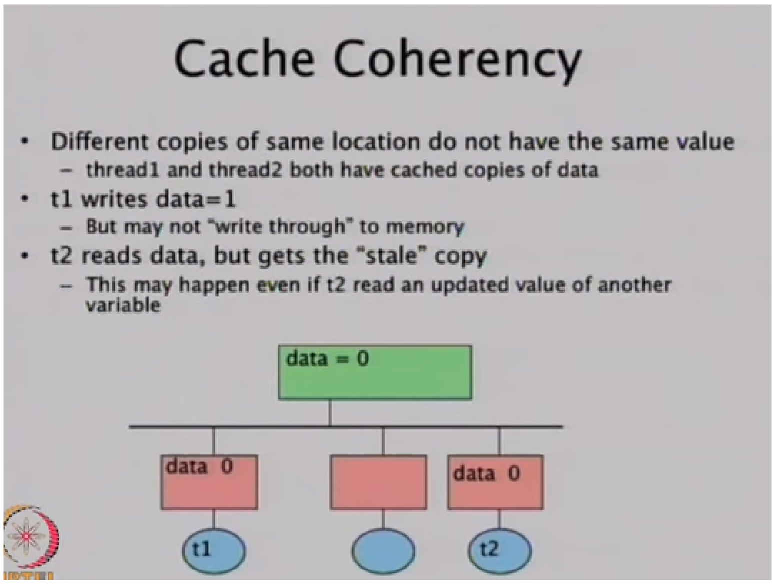
For example and all of them had that thing that you have to depend on something else to get to this the next step and so there are so many dependency is number of steps that that automatically do this in log time. LP that would be very efficient in fact the problem is that I got a bunch of planes alright the feasibility version of the problem only says I have got a bunch of places does their exist in every points which is in positive half price or all of the price or negative half space.

That is the another version the full linear programming says that I want linear combination that those objective functions and I want to know in this feasible space is this objective function maximized okay. That is as much we are going to talk about complexity and parallel computation models am going to switch to memory consistency models and then want to programming design paradigms any question on stop of discussion so far not in NC.

They can be done in NC time so why do say not NC because binary search is not NC binary search itself although realize that when it is MC that does not always mean very much for all problems. The sequential is order N right N steps in parallel we can do in log N time prefix M or we will do those things so we will go through standard parallel algorithm that you will use as sub routine in a variety of problems and but we will get to that in may be a month from now or may be less okay.

NP is NP no I have just drawing analogy that every definition you would have seen in a MP is just you replace some key words and got this new set of definitions.

(Refer Slide Time: 26:03)

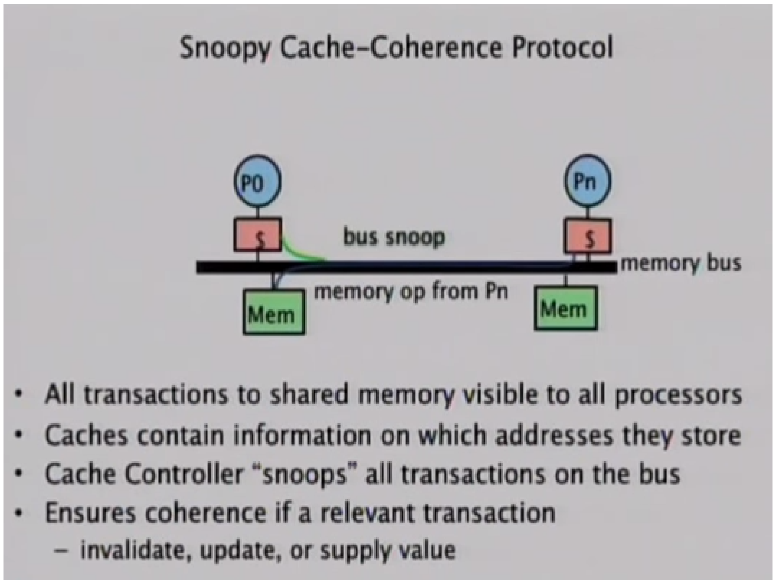


And you all probably all know them this is getting almost the notion of cache coherence right it is basically on hardware level typically implemented already discussed this in short before also there are N number of processors each in this case the are written as T1, T2 etc threads so to speak may or may not be running on the N processor and they have their local data stores may be caches and the shared data store the green color one which is the shared memory.

And you are going to read and write everything is just going to stay in cache until it needs to evicted right. And so you write something to cache other processor writes something to its cache and you read something back after that other processor writes is still get your data one okay. So you do not want that happening that is why cache coherence is written into hardware implementations and the any honest to goodness machine will have some level of cache coherence.

And that essentially means keeping the cache state consistent it is another way of saying that you basically hide the cache from the program. The program should not have to know that the data came from the cache it should appear to the program that the data came from the shared memory main menu okay. This is typically going to be implemented using validate type protocols that we sort (()) (27:51) maintained.

(Refer Slide Time: 28:02)



So is an example that you would also potentially have some snoopy cache which is little bit more efficient but much more hardware required where you keep figuring out what cache information is flowing on the bus to update yourself without somebody having to tell you that other fellow has written a new information and that data item. So we are going to take all them grand right it depends on right back and write through policies.

Snoopy cache it will only work when you write to cache as well as into the main menu all of this is typically at granularity of one cache line. So if you see a cache line going on the bus that you also have you are going to update you cache column. Because that cache data being written is more being than yours okay so the context is one memory item typically a cache line may even be one bytes 4 bytes whatever it depends on the architecture but we will just say a cache line and that is from the hardware point of view.

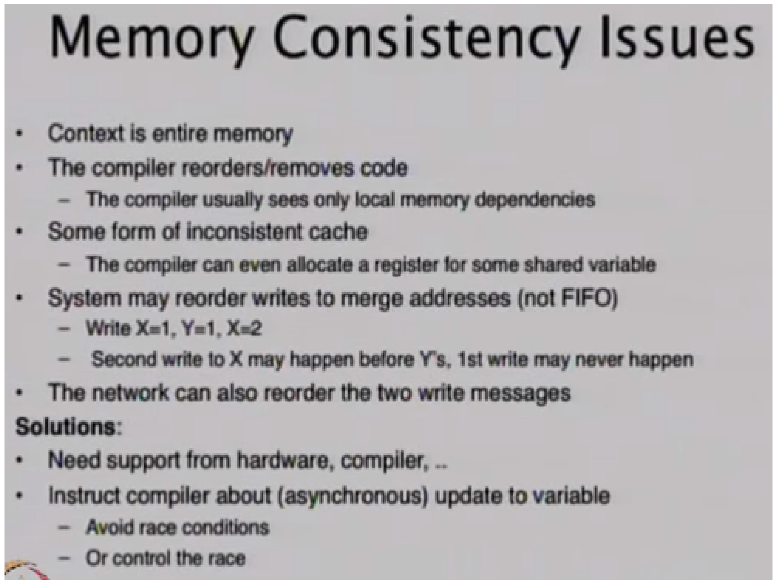
So this is the hardware centric view what we need from the program point of view we are writing a program and we say okay the hardware make sure that there is cache coherence meaning if I write X then the other processor will know that value of X okay if that we are also have it in cache. If that cache does not care it is not going to look for that X is that enough that is sufficient.

Do we need a more global view okay the entire memory state is needs to be somehow consistent meaning that sorry has in that variable Y and variable X a completely different location may depend on each other pointer is one possible the other is a pure logic right I have read X before I do not have it any more but I have use that in Y in computation of Y. And so from hardware point of view is just matching addresses from program point of view it is overall state of program.

I should say that is all that the hardware is going to provide that is not enough from the program point of view because you are not going to read that you never intending to read that guys written value because you might have read before that long time back. So that overall consistency has to involve the programmer over there in that paradigm where only thing that they hardware provides is address to address coherence into cache and then the programmer takes care of today's conditions that is one model.

Either is that the underline system gives you a better view of consistency a better consistency of memory okay.

(Refer Slide Time: 32:43)

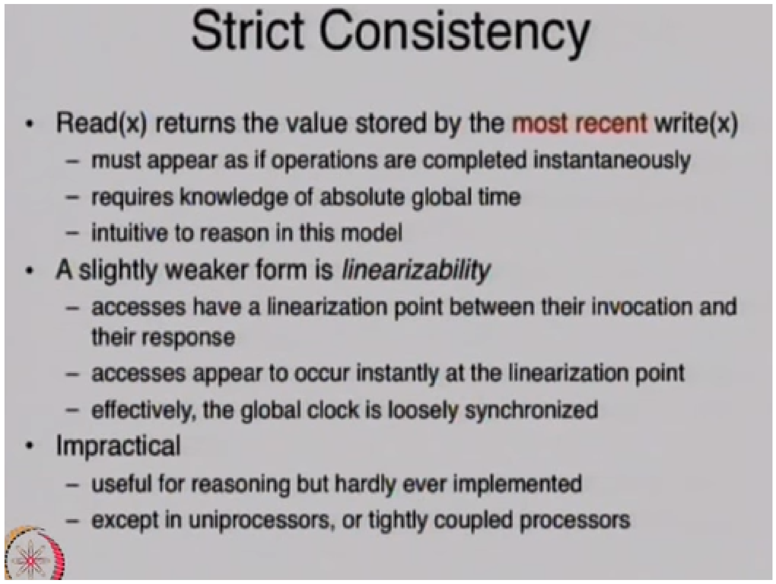


So in this case the context is the entire memory and we want somehow your programming interface the compiler the hardware together giving you this view right a contract to the programmer that says that when you access things in that way this much is guarantee to have and these kinds of things are very easy to do in the processor level. Because there is only one fellow interested in anything and whatever may be along the way can all be reordered in the same way so that you get everything pipeline in same fashion okay.

But as soon as you have got multiple processors even if they are talking to the same shared memory there is typically difference in timing at the level of the logic and sometimes may be actually physically be a shared memory. Sometime they may implement shared memory on top of some distributed memory network. And so in that case when it is very clear that latency from some rides is going to be much higher than some other writes okay.

So it is easily going to get reordered by the system and you need to make sure that when the program logic is seen some consistency that is expected by a programmer is provided. So we look at some of these definition of some of the consistency it might be able to do that but first we have to determine what the contract is that the compiler is making with the program or the programmer right.

(Refer Slide Time: 34:50)



In the unique processor domain it makes lot of sense write something I write it again will get rabble in second one but when I write two things one after the other somebody else may read this state in the middle when I cannot simply say I cannot I do not worry about this and all okay so other programs may be it may or may not be the same program running elsewhere right.

It is completely a different program accessing the same piece of memory so we have you have no idea all that is going to behave okay and so the most stringent model of consistency memory consistency is called the strict consistency model where basically have a global clock and any time somebody says read everybody stop doing everything else that read gets completed when they start effectively right means everything happens is stringent.

The consistency definition is simply contract with the program which means that the underline this hardware software thing that the program is sitting on top of it is giving you this behavior is giving you that apparent behavior what a how it is managing me it is business. This says what kind of different things so there are two partial contract right the programmer and whoever it provided in this case the hardware compiler together.

And we are trying to understand what the program wants to see okay and then see whether the hardware wants to provide and then balance it out. The strict consistency is what the programmer what like to see so is everything is globally ordered so I know when I am reading that other guys read happened 5 millisecond earlier and so it happened before it not race conditions because I know timings.

The hardware is going to somehow order everything is done instantly and there is this additional kind of hidden rule that instantly may not be the same instant for two operations. So the effectively right so program is going to see some behavior from the memory when it reads it see something basically that is the response that it gets when it reads something what is the value it is going to get right and so there is a read and the write and you were not going to allow them to happen instantly at the same time.

Each will happen instantly but one have to happen before here that is strict consistency model those details are not necessary part of the consistency model it just says that when I make something at a given instant of time we all know the time and that guy makes some other request at a given instant of time you cannot make request at the same exact same instant of time the whoever made earlier is going to go earlier is going to start and finish before the other one starts it is atomic.

Because everything is atomic order is given no order known because you have the clock everybody knows the clock so you can see what time you have made this request okay. There is no actual implementation of the model you would use that implementation by knowing that I have made this request at certain of time twenty forth instruction and that guy and then I can write even the instruction number where I am making this access read or write and then I will know whether that guy got it first or got it later.

You can make is non-deterministic but you can also make it deterministic given this model okay you can have the checks on time stand everybody is access to the time stub every step is time stub. They are so you can think also again we are probably to worried about the implementation part of it the issue is of an in fact that may be one of the reason when nobody implements it because it is extremely challenging to in fact providing global clock is maximum impossible okay in a distributed circle.

But you can simply say I have got time stands 5 steps and your processor number is sub time step so you go you log go first you log go second you will always go so everything is disambiguated so that is one way of doing it okay. And then many other who knows what different contact might prefer a slightly the weaker form of this strict consistency is a again it appears atomic but that global time stand is only approximate wage.

Meaning that I do something here I do something here else over here and between these two times access right we will do access over here sometime later and there is some time stand for this which is definitely more than this and my access is sometime send is going to get effectively the what is going to be visible in all the program system is whole will effectively done some point of time atomically.

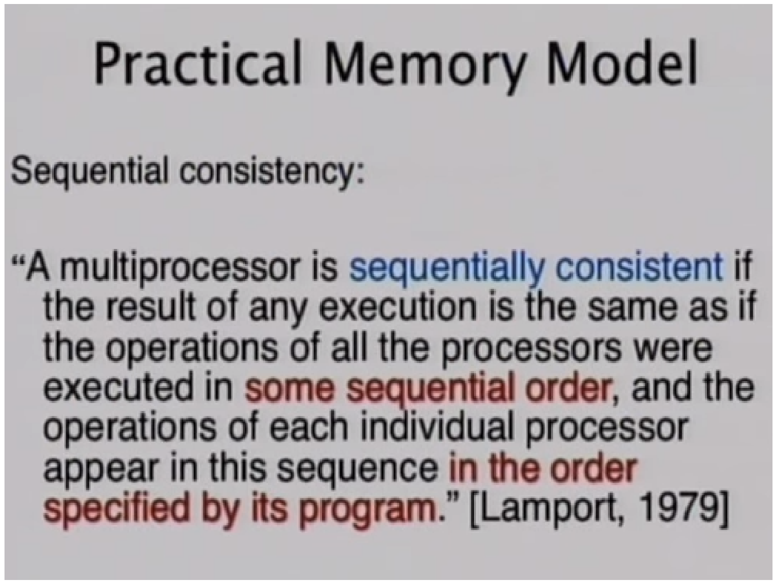
But based on this approximate time stand so we are not in lock step fashion we do not know exact time stand in mean time but we cannot be off by two much but next access has to have a bigger time stand we will take some examples. There is basically we will the take examples especially that we will use the global clock is based on the original statements of the offer basically says the global clock is not required to fully synchronized.

That means is that we do not know the exact time when the process said I want to do this we can order them in all the models even the weaker models but we are going to somehow order than but just based on their relative position in the program but with respect to the loosely synchronized clock we do not want to know the precise time at which the instant so this is somewhat similar to one giving you the fully deterministic sequence one giving you non deterministic sequence except it is still it is atomic.

So every access appears to happen instantaneous but at what time it appears to have instantaneously is can vary by small amount delta lack of synchronization is going to allow okay. So those are impractical models when I was because of the expense in implementing this synchronization both of them expect some notion of this global time I need to be able to know that this time is so many events alright.

And they have generally being used to prove theorems about solvability of some programs of problem and the like and the lower bounds certain things but you will see that the actual consistency model provided by real implementations much loser that either of this.

(Refer Slide Time: 46:02)



And the primary thing that is the notion of clock so this is probably the first relevant still the strict test consistency model which is called the sequential causes alright and was first given by LAMPORT in 1979 and it says that and this is I think we something discussed shortly earlier also briefly says that in multi-processor system in a many processors talking to the memory.

It the interface all the programming paradigm is sequentially consistent if it is equivalent in some linear order of operations okay. But that linear ordering must respect the ordering of each individual program right take the subset of the so all the operations are going to be ordered and as long as that order can be generated in sequentially consistent we take the subset of that of operations for each thread or processor or originally send processor then it is going to be the same as that processors request accesses.

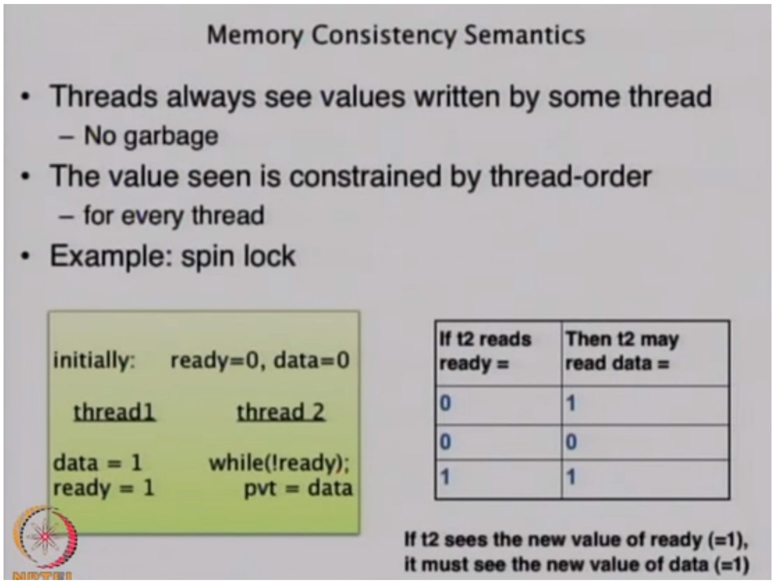
You cannot reverse it right you do not internally make them still it is not read as started the year and ended here because that the original the concurrent model where I do not know when it is going to finish I give read here and at some point read as happened and that over with lots of other reads in that gets if you do that then everything as to be decade by locks in other like so that is the full concurrent model.

Here in sequentially consistent model it is the behavior that a program sees must be the same as the behavior seen by some sequential program given that order that this overall system saw seen by that sequential program right meaning that the if I sorry so atomies it is effectively providing atomicity to each access right so each read or write separately they will be internally reality right.

But it should be effective read right I will get you can example right now previous slide or when you are talking about open MP which had a notion of loosely synchronized clock here there is not notion of clock the ordering can be ordered anywhere right you are basically saying I have these thing here these things here just merge them in any fashion I like. If you take out if you take one processor access then they have to be ordered in the global order.

So let me restate you create a global order right all accesses are there in global order all reads and writes now you only consider the accesses made by thread one in this global order they must appear in the same order in which thread made them okay that is all that is needed it seems reasonable turns out still be hard to them and it is performance where it is implemented it is performance is not that great yes.

(Refer Slide Time: 51:04)



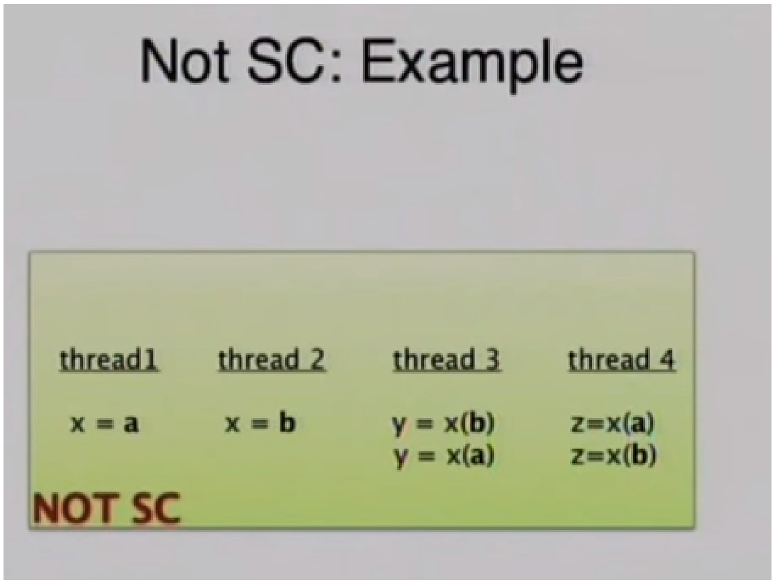
So here is an example got I have got two data items ready is 0 data is 0 when we begin right and two steps have started thread one and thread two. And thread one says data = 1 and then ready = 1 and thread 2 mean while is reading ready many times as long as it remains 0 as soon as it is sees it is 1 it comes out and it comes out and let us not supposed to tended like that there is a semi colon it keeps looping so that tab is extra.

And then it says private = data right so when it thinks that the data is ready it makes a copy of it okay so what are we expecting in terms of sequential causes basically means that ready is value data must be before it because in its order data is happened before it. So the program expects when I had seen that ready is one I am guarantee that data will also is 1 okay. I do not care when data got return.

I do not care when my access came in my read value is came in I do not know if a given read for the ready and a given write for ready where ordered in the certain fashion when it was looping I do not care about that but if I was doing strict consistency there as to be order among them okay all those reads have to be done at the time that the global clock set that it supposed to be done.

Here we do not care you can take all my reads and order them after the data is have has been written before the data has been written or after the ready has been written okay. I do not care when you did it you can order it anywhere okay so here it is the little table that shows and if T2 read reads 0 then data might B1 or 0 if T2 reads 1 data is definitely 1 okay.

(Refer Slide Time: 54:21)



Here is an another example right whatever so I have got 4 threads, thread 1 writes the value A to variable X at this X threads 2 writes B I do not know anything about global clock right. So I do not know whether X = A happen first or X = B happen first if I was asking for strict consistency I would know which has to be first okay previous one we were are not enforcing anything in previous case that is what the model was providing to us.

We were expecting back to the previous we were expecting that if we are reading ready = 1 by that time data must be valid because I am now going to read data then you are expecting the behave to be correct okay. If it is not sequentially consistent will it may behave because if I am ready there may be done in a different order it is possible that data is far away and writing to text longer then writing to ready right.

So ready got done data never got done in this case that is where am going thread 1 read wrote the value A into variable X thread two wrote the variable B into X thread 3 is going to read X twice and thread 4 is also going to read X twice okay and the bracket shows what the value it got when the value is read thread 3 read B the first time and A the second time thread 4 read A the first time and B the second time.

Because may be the thread A had was closer to B closer to the second write or whatever okay now any order you produce can provide what they saw so that means not there is no ordering which is consistent with the responses that everybody got is not sequentially consistent. If they saw this B okay in turns out as I said and I am going to stop now because we are running out of time okay sequentially consistent is meaningful you can write program with a lot of guarantee on when raises will not happen but it is surprisingly hard to implement and nobody does it either.

Which means that we will be looking at even more lack X models of consistency which open MP for example implements something we said is weaker consistency model it is something called weak and it is weak then the weaker and then that even less strong consistency models that actual multi-processor implementation use which case they basically shifting more of more is responsibility to do the program work to make sure that all of the race conditions or either avoided or controlled at the cost of during better performance.

The idea is the programmer knows best of what where the rise is likely to occur but it is the same story it is enough to be even you can use to tie others or hand yourself okay alright so we will stop here now.