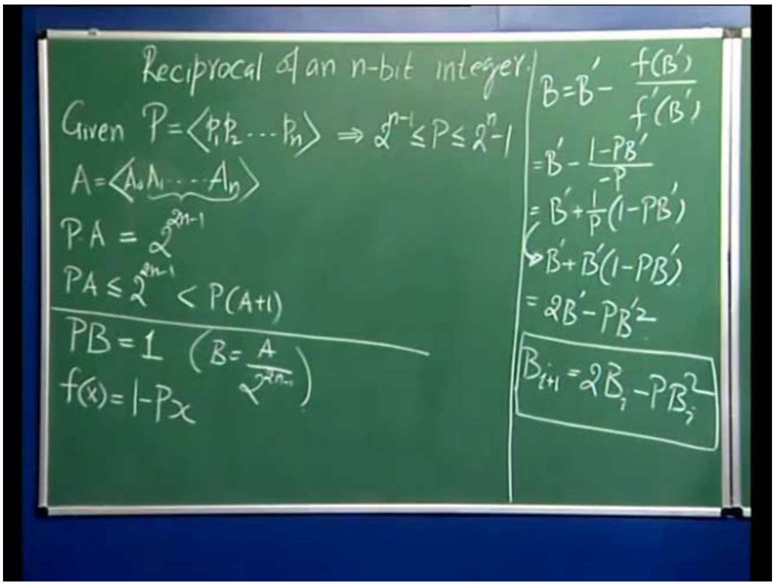
Computers Algorithms-2

Prof. Dr. Shashank K. Mehta
Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur

Lecture - 17
Integer-Polynomial Ops I

In this couple of lectures, we are going to discuss algorithms related to integers. Our goal is not to provide an efficient algorithm instead what we will try to establish is that the difficulty of multiplying two integers or computing reciprocal of an integer or computing the square of an integer, all these problems have same difficulty. So, today we will begin with a problem of computing the reciprocal of an n bit integer. We will try to show that the difficulty is same as that of multiplying two numbers of size n bits.

(Refer slide Time: 01:09)



Reciprocal of an n-bit integer.

Given $P = \langle P_1 P_2 \cdots P_n \rangle \Rightarrow 2^{n-1} \le P \le 2^n - 1$

$A = \langle \underbrace{A_0 A_1 \cdots A_n} \rangle$

$P \cdot A = 2^{2n-1}$

$PA \le 2^{2n-1} < P(A+1)$

$PB = 1 \quad \left(B = \frac{A}{2^{2n-1}}\right)$

$f(x) = 1 - Px$

$B = B' - \frac{f(B')}{f'(B')}$

$= B' - \frac{1 - PB'}{-P}$

$= B' + \frac{1}{P}(1 - PB')$

$\to B' + B'(1 - PB')$

$= 2B' - PB'^2$

$\boxed{B_{i+1} = 2B_i - PB_i^2}$

So, let us suppose we have an integer P of n bits, which I am going to denote in this fashion. So, this is the least significant bit, this is the most significant bit. We would like to compute another n bit number say A, such their product of the 2 numbers that is P times A should be 2 to the power 2 n minus 1. Now, notice that this number is between 2 power n minus 1 and 2 power n, minus 1. Now, the reason for solving this problem is that in general the reciprocal of a number is not going to be representable infinite number of bits and in case it has a finite number of bit representations it may not be small. Our goal is to get the accuracy of n bits in the reciprocal, so we are representing it in this fashion now.

But there is a little bit of A special case that we have to address, what happens is that when P is exactly 2 to the power n minus 1. In that case, our A will turn out to be 2 power n and that requires n plus 1 bits, beyond that point for every other value of P, we will have a number which will be smaller than this and we will not need an extra bit. But to cover that case we will grant our self n plus 1 bit, so we will say we will have A 1 A 2. So, in other words we will have up to n plus 1 bits, the main information will be always in these n bits and only in very special case when P is exactly 2 power n minus 1 the number will be 1 0 0 0 0 0.

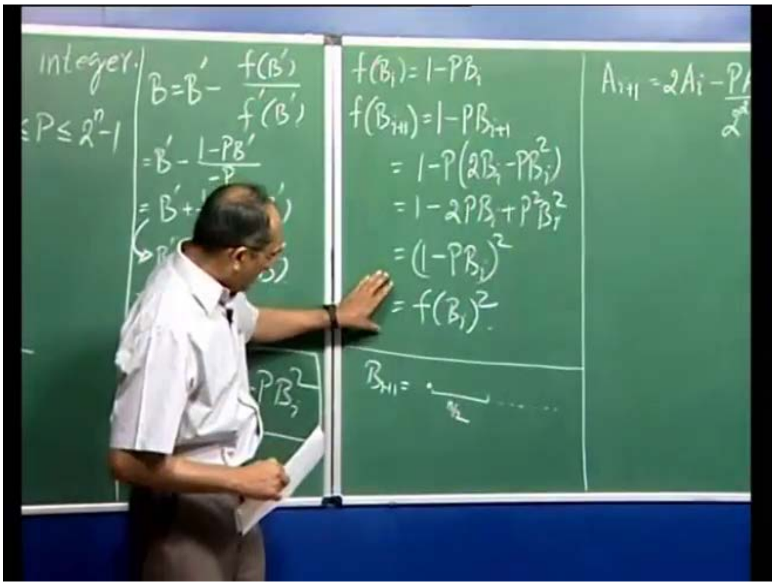
So, we will now try to address, we will try to decide an algorithm, which computes A such that this is as close to this as possible. Remember that this is not exact number, so we hope that, so we will compute P A less than equal to 2 power 2 n minus 1, but strictly less than P A plus 1, so that will be our best bit for A. Now, our approach will be iterative what we will do is we will iteratively improve our solution starting from in an approximate solution and we will iteratively improve.

Now, before I get back to this problem, let us take A special case of P B equal to 1, in other words my B that is my B is same as A by 2 power 2 n minus 1 this is just for the ease of our discussion. I am assuming this number B. So, this is not an integer now we are just trying to approximate the best possible value of B that we can, now the process of approximation will try to minimize the error. So, let us define an error function f of x to be 1 minus P x, so if we have exact solution then of course, the error will be 0.

In order to decide an iterative procedure, so that our error keeps on decreasing, we will just use A the Newton graham’s method and what it says is that if we have B prime as previous approximation to this then we will compute B by this method. We are just taking f x divided by f prime x and then we are computing at B prime. So, let us just take the look at this, this turns out to be B prime minus 1 minus P B prime that is our function f at B prime and its derivative is minus P, this is such as 1 over P 1 minus P B prime.

Unfortunately we do not know 1 over P that is precisely what we are trying to compute. So, we will plug in the approximation that we have and that will make A B prime plus B prime, so we will not say this is equal, but we will transform this into this and that is 2 B prime minus P B prime square. So, let us say we will use the iteration B let us i plus 1 equal to 2 B i minus P B i square.

(Refer slide Time: 08:36)

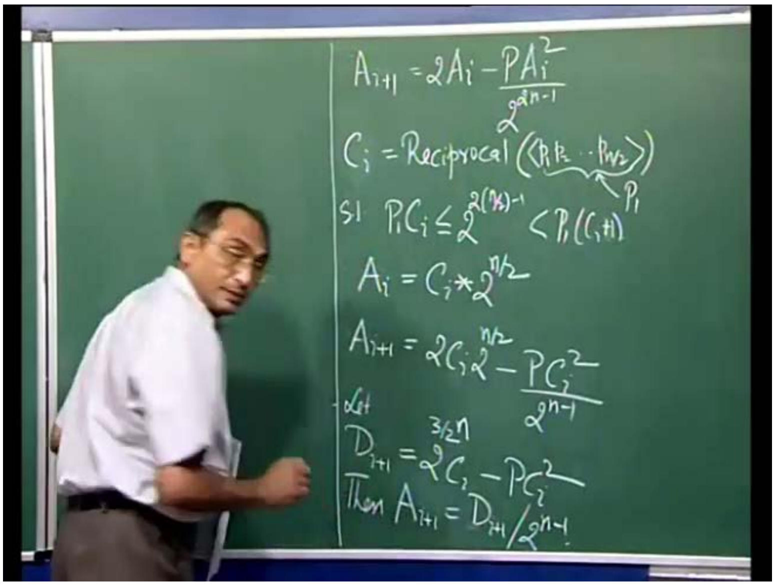


So, let us try an estimate the error in one iteration. So, the error in the previous step P B i was 1 minus P B i and the new error f of P B i plus 1 is 1 minus P B i plus 1, which is 1 minus P. We plug in this value 2 B i minus P B i square. That is 1 minus 2 P B i plus P square B i square and that is 1 minus P B i square which is f B i square. Now, what this indicates is that in case the error in the previous step, in the i th step was a fraction of 1 then the accuracy in the new step in terms of number of digits of the decimal will double. So, it exponentially approaches the accurate value and this is why we will go with this approximation.

Now, let us start to plug in the value of B in terms of A, notice that B can be replaced by A by 2 power n minus 1 because that would give us recurrence relation for A. So, we have the recurrence relation, A i plus 1. All we have done is we have multiplied to the 2 n minus on both sides that makes it 2 A i minus P A i square by 2 power 2 n minus 1 that becomes the recurrence in terms of A’s. Now, notice that if this was the last A value suppose, this was the last value and let us say that value or may be in terms of B.

Let us just say in terms of B, B i plus 1 is decimal. We have let us say n by 2 bits of accurate bits here and the other n by 2 are not reliable. Then our argument here says that in one iteration all this n bits will be accurate almost. So, in order to compute my mistake this is we are taking about B i, in B i we are going to have accuracy of n by 2. In order to compute B i we should, therefore not utilize all the n bits of P instead we should just use the first n by 2 bits of P, because our concern is to that an accurate first n by 2 bits. So, what we will do is we will recursively compute A I, but this time we will use only first n by 2 bits of P.

(Refer Slide Time: 13:01)



$$A_{i+1} = 2A_i - \frac{PA_i^2}{2^{2n-1}}$$

$$C_i = \text{Reciprocal}(\langle \underbrace{P_1 P_2 \cdots P_{n/2}}_{P_1} \rangle)$$

$$\text{s.t. } P_1 C_i \le 2^{2(n/2)-1} < P_1(C_i+1)$$

$$A_i = C_i * 2^{n/2}$$

$$A_{i+1} = 2C_i 2^{n/2} - \frac{PC_i^2}{2^{n-1}}$$

Let

$$D_{i+1} = 2^{3/2 n} C_i - PC_i^2$$

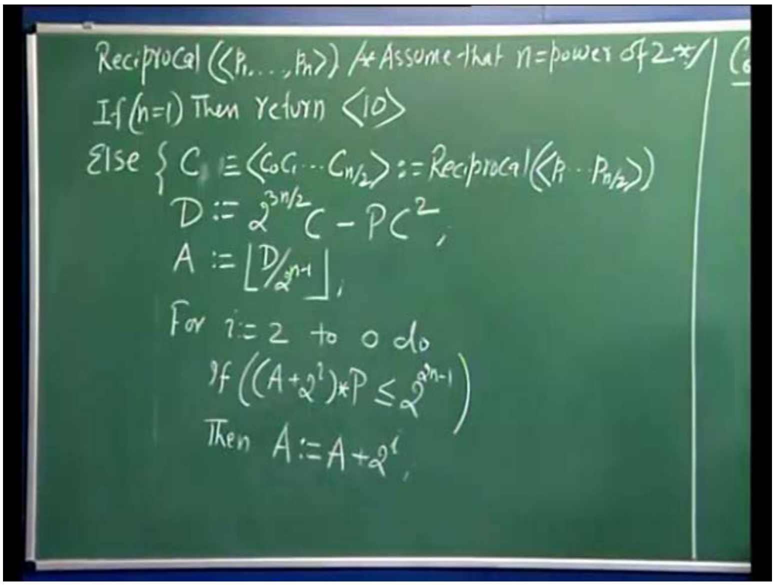
Then $A_{i+1} = D_{i+1}/2^{n-1}$

So, let us say we have A i as reciprocal of now reciprocal is the procedure which we will write and that will compute the n by 2 bit reciprocal of the n by 2 bit input p 1 p 2 p n by 2 such that and let us call this number P 1, we want that P 1. Let us say we first label this as some other number C i, so we want P 1 C i to be less than or equal to 2 to the power 2 n by 2 minus 1 and less than P 1 C i minus 1. This is the same kind of reciprocation where we are only dealing with n by 2 bit numbers and the outcome of this call is in n by 2 number C i such that this is the condition it satisfy.

In that case we will then make A i to be C i into 2 to the power n by 2 will extend it to an n bit number, notice that I got this accurately and then I am just putting 0s because I do not care what there is. Our recursion says that all we need is, accuracy of n by 2 bits here to get an n bit accuracy over here. So, if we have C i which is due to a previous call over an n by 2 no significant bits of P then we will take C i upend another n by 2 bits 0s to it to lower less significant side. Then we will use our recurrence to get the desired number of n bits. So, that was an error here we should have a plus 1 here. Now, let us try to plug in the value of in this A i in this expression, we are going to get A i plus 1 as 2 C i 2 power n by 2 minus P C i square over 2 power n minus 1, 1 n will cancelled by the extra 2 power n by 2 square which is 2 power n.

Now, let us define let D i plus 1 be 2 to the power 3 by 2 C i minus P C i square then A i plus 1 is nothing, but D i plus 1 divided by 2 power n minus 1 just that I have multiplied the entire expression the 2 power n minus 1 to define D i plus 1. And hence this is A i plus 1 is D i plus 1 divided by 2 power n minus 1. So, now let us try to write the procedure which is very short one. Let us call it reciprocal of an n bit number, now first we check if notice that when we have an n bit number we mean that p 1 is 1, this is 3 by 2 n.

(Refer Slide Time: 18:28)



```
Reciprocal(<$p_1, \ldots, p_n$>) /* Assume that n = power of 2 */
If (n=1) Then return <10>
Else { C ≡ <$c_0 c_1 \cdots c_{n/2}$> := Reciprocal(<$p_1 \cdots p_{n/2}$>)
    D := $2^{3n/2} C - PC^2$;
    A := $\lfloor D/2^{n-1} \rfloor$;
    For i := 2 to 0 do
        If ($(A+2^i) * P \le 2^{2n-1}$)
        Then A := $A + 2^i$;
```

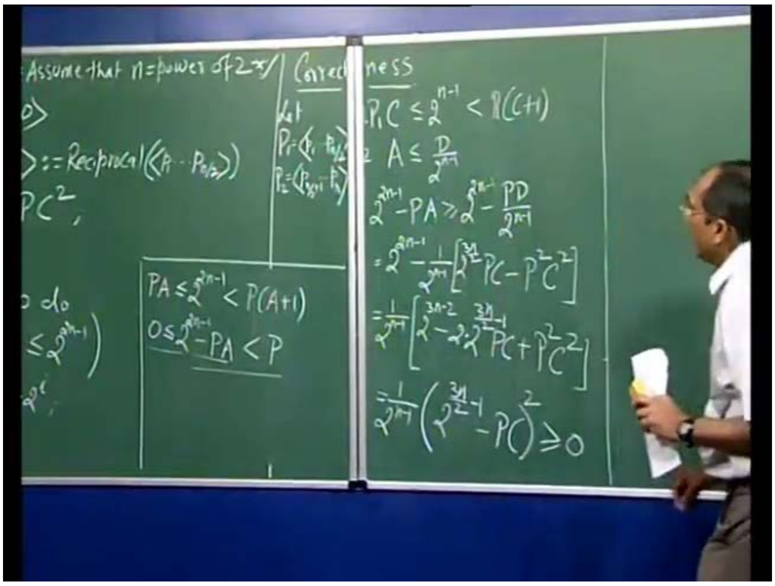
Coming back to our procedure, when we say it is an n bit number we mean the most significant bit is 1 because by padding 0s were not really getting any new information, so it is assumed always that this is 1. Now, in case n is 1 then clearly we asked to return 2 because 1 into 2 is 2 to the power 2 n minus 1 which is 2 power 1 that is 2. Now, this is our base case and now we are ready, now here we are going to assume I forgot to point out. I assume that n is a power of 2, I will later on address the general case, that is not very difficult to handle, so we will assume that we have power of the n is 1 2 4 8 16 and so on. Now, in this case let us compute C 1 or that is maybe I could just say C 0 C 1 through C n by 2 to be equal to reciprocal of the most significant n by 2 bits of p.

Now, let us compute, we do not need to have a subscript C 1, it simply says C and then we define D as we have done there, that will be 2 power 3 n by 2 C minus P C square. We will define A to be the floor of D divided by 2 power n minus 1 now this is our desired A, so that is it. So, ideally I should return, but later on I will show that there is something required here and that is actually a very simple step effectively what we will show is that this A is not the desired A. But it may be off by 4, it may be the desired A or may be minus 1 or minus 2 or minus 3 or minus 4, so it might be little bit short and the next step is essentially to fix it explicitly.

So, all we are doing is we will say for i equal to 2 0 do if A plus 2 power i times p is still less than or equal to 2 to the power 2 n minus 1 then is A plus 2 to the power i. So, what we try to do is check whether you can add 4 in it in the first right way, if not the second alteration will try to check whether it can be added 2 and yes or no, then we will check whether it can be added one? This way we can actually check whether A or A plus 1 or A plus 2 or A plus 3 or A plus 4, either of the satisfies this condition. We know that A plus P will always be exceeding the limit.

Hence, we do not need to worry although this procedure checks up to P, so this is our complete procedure. Next, I am going to show the correctness of this procedure, so let us try to look at key steps. So, first thing I am going to do is will just try to point out that in recursion we have made a call for C. So, of course, base case is true for we do not have to make any difference and we are going to prove by induction, so from induction I have to this is can take this is correct as expected by this procedure.

(Refer Slide Time: 24:06)



Assume that n = power of 2
::= Reciprocal(⟨P₁ · P_{n/2}⟩)
PC²,
Correctness
Let
P₁ = ⟨P₁ ··· P_{n/2}⟩
P₂ = ⟨P_{n/2+1} ··· P_n⟩
PA ≤ 2^{2n-1} < P(A+1)
0 ≤ 2^{2n-1} − PA < P
P₁C ≤ 2^{n-1} < P₁(C+1)
A ≤ D/2^{n-1}
2^{2n-1} − PA ≥ 2^{2n-1} − PD/2^{n-1}
= 2^{2n-1} − 1/2^{n-1} [2^{3n/2} PC − P²C²]
= 1/2^{n-1} [2^{3n-2} − 2·2^{3n/2-1} PC + P²C²]
= 1/2^{n-1} (2^{3n/2-1} − PC)² ≥ 0

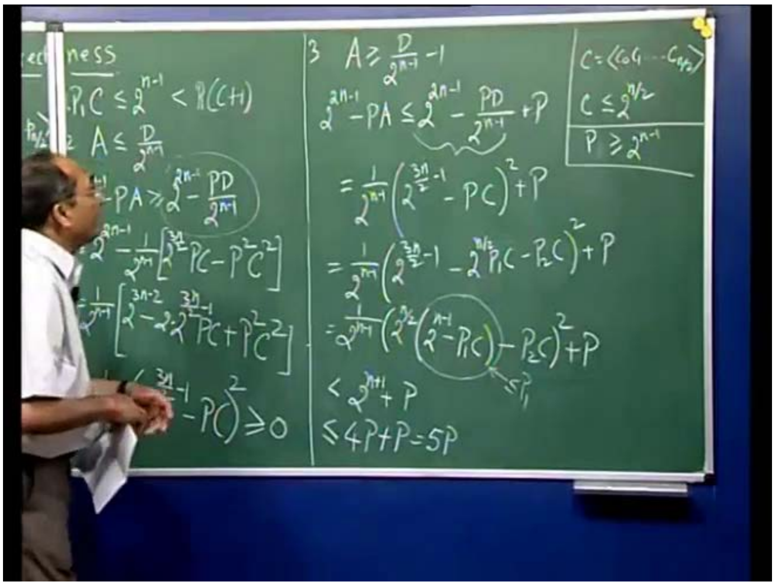
So, based on that what we know is that P 1 C is, so let p 1 be the n by 2 number n by 2 bit number p n by 2 this is the and I will also denote the next n by 2 bits as p n by 2 plus 1 through p n from splitting the p into this 2 parts. Then, P 1 C must be less than equal to 2 to the power n minus 1 and should be strictly less than P 1 C plus 1; this is something we can take for granted from induction hypothecs, so this is step 1. Step 2, let us try to compute following expression, first of all let me just point out that A is less than or equal to D divided by 2 power n minus 1. Notice that we chose A to be that floor of this by 2 power n minus 1.

So, we know this and the our goal, just go back and recollect that our goal is to show P A is less than equal to 2 power 2 n minus 1 strictly less than P A plus 1. Of course, we will not be able to show this type, we will show slightly with a result, but let us just keep the goal and this says that we want to show 2 power 2 n minus 1 minus P A is less is positive, but less than p, if I subtract P A from all the three sides we get this. So, I need to show these two inequalities or something close to it, so let us try to get the expression for 2 power 2 n minus 1 minus P A this is greater than equal to 2 power 2 n minus 1 minus P D by 2 power n minus 1, the reason is this in equality.

Let us plug in the value of D in this, so this thing is 2 power 2 n minus 1 minus and our D is this expression. So, we have 1 over 2 power n minus 1 and we get 2 power 3 n by 2 P C minus P square C square and is 1 over 2 power n minus 1, 2 power 3 n minus 2 goes up 3 minus 2 power 3 n by 2 P C plus P square C square and that is 1 by 2 power n minus 1, 2 power 3 n by 2 minus 1. Now, notice that I can replace this expression by minus 1 and put 2 there, I can express this by 2 time, I just divide this term by 2 and put A 2 there.

So, this looks like a whole square of this expression and this is clearly non negative. So, what we have found is where indeed 2 to power 2 n minus 1 minus P A is non-negative that is, this side of the inequality as we must have. Now, let us look at the second one, next we are going to show this side of inequality and for that I am going to use the fact that A was actually lower of D over 2 power n minus 1.

(Refer Slide Time: 29:23)



ness

$P_1C \leq 2^{n-1} < P_1(C+1)$

2 $A \leq \frac{D}{2^{n-1}}$

$-PA \geq 2^{2n-1} - \frac{PD}{2^{n-1}}$

$2^{2n-1} - \frac{1}{2^{n-1}}\left[2^{\frac{3n}{2}}PC - P^2C^2\right]$

$\frac{1}{2^{n-1}}\left[2^{3n-2} - 2\cdot 2^{\frac{3n}{2}-1}PC + P^2C^2\right]$

$\left(2^{\frac{3n}{2}-1} - PC\right)^2 \geq 0$

3 $A \geq \frac{D}{2^{n-1}} - 1$

$2^{2n-1} - PA \leq \underbrace{2^{2n-1} - \frac{PD}{2^{n-1}}} + P$

$= \frac{1}{2^{n-1}}\left(2^{\frac{3n}{2}-1} - PC\right)^2 + P$

$= \frac{1}{2^{n-1}}\left(2^{\frac{3n}{2}-1} - 2^{n/2}P_1C - P_2C\right)^2 + P$

$= \frac{1}{2^{n-1}}\left(2^{n/2}\left(2^{n-1} - P_1C\right) - P_2C\right)^2 + P$ $\leq P_1$

$< 2^{n+1} + P$

$\leq 4P + P = 5P$

$C = \langle C_0 C_1 \cdots C_{n/2} \rangle$

$C \leq 2^{n/2}$

$P \geq 2^{n-1}$

So, we know that A is actually greater than or equal to D over 2 power n minus 1 minus 1 because of the fact that we took D over 2 power n minus 1 4 as A. Once again I am going to just plug in the value and try to compute 2 to the power 2 n minus 1 minus P A is greater than equal to that could be less than or equal to 2 power 2 n minus 1 minus P of P D divided by 2 power n minus 1 plus P. Now, this expression we have already computed this is this expression and that was exactly this. So, I can plug in that and we get 1 over 2 power n minus 1 2 to the power 3 n by 2 minus 1 minus P C square plus P.

Now, what we need to do is, try to express this ideally I would have like this think this less than equal to P. But, this is not 0, so I need to reformulate this expression so that we can rewrite this in terms of P. So, I am going to spilt this P into P 1 and P 2, we have 2 power n minus 1 2 power 3 n by 2 minus 1 minus. Notice that P is nothing, but 2 power n by 2 P 1 plus P 2 2 power n by 2 P 1 plus P 2 C. So, I am going to just expand this and that will be 2 power n by 2 P 1 C minus P 2 C square plus P. This is 1 over 2 power n minus 1 2 power n by 2 power n minus 1 minus P 1 p minus P 2 C square plus p the purpose of splitting this was to be able to use this in equality.

Notice that p 1 C is less than equal to 2 power n minus 1 2 power n minus 1. So, the difference 2 to power n minus 1 minus P 1 C is less than or equal to P 1, so this term is less than or equal to P 1 which is A an n by 2 bit number. Hence, it less than 2 power n by 2 put together these 2, the number is less than 2 to the power n. Now, on this side, this is less than or equal to 2 the power n by 2 remember this is A n plus let me just remind you. That C was C 0 C 1 C n by 2 where this will be 1 only when all these are 0 remember that was only A special case else this is 0. So, we knew that C was always less than or equal to 2 th power n by 2 p 2.

On the other hand is strictly less than or 2 power n by 2 where the product is again less than 2 th power n. So, notice that we have a number which is less than 2 power n and n, a number which is also less than 2 power n, their differences also less than 2 power n. So, we can say this is less than or equal to 2 to the power 2 n divided by 2 power n minus 1 is 2 to the power n plus 1 plus P is not that right. So, we have minus 2 to get an expression for this an upper bound in terms of 2 power n plus 1. Now, we know that the most significant bit of P namely P 1 was 1. So, we also have a lower bound, we know that P 1 is greater than equal to 2 power n minus 1, it can be smaller than this P 1, sorry P, P is greater than equal to 2 power n minus 1.

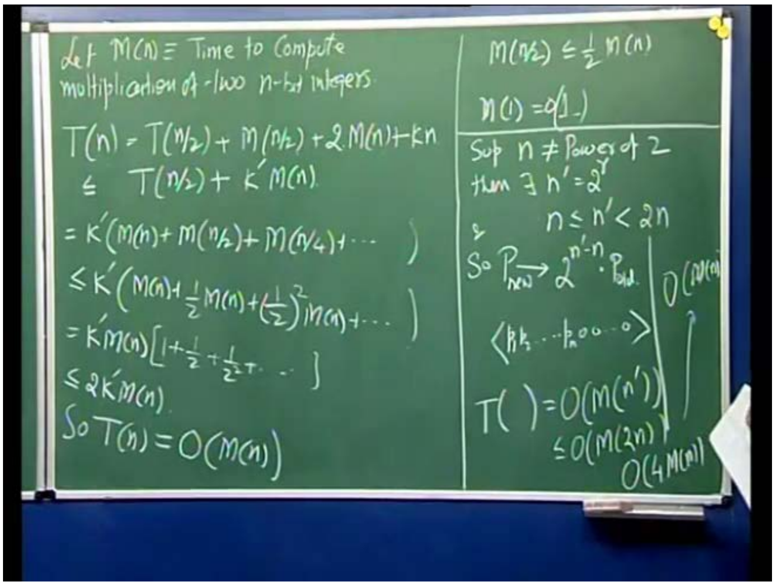
Using this inequality I can bind further and say this has to be less than equal to 4 P. So, this is less than equal to 4 P plus P this is 5 P, actually this is strictly less, the reason is in here. This was strictly less than n by 2 and here this was strictly less than n by 2, so both these terms were less than it is less than 2 power n and that difference is also strictly less than 2 power n. So, we can actually put a strong inequality here and this is bound by there. So, what we have concluded is that instead of this quantity being less than equal to P, we are able to show it is less than equal to or strictly less than 5 or less than equal to 4 P. So, we have ended up showing that P A is less than equal to 2 power 2 n minus 1.

I will actually, this is what we have computed in this step. In this step, I am calling it A actual and that is less than P A actual plus five term was 5 P, in other words my desired A might be A actual or A actual plus 1 or plus 2 or plus 3 or plus 4. Now, in order to fix that, we explicitly go about if it can accommodate all the four in one. Though, we will replace A by A plus 4 and to check whether it that the term rated be just less than or equal and then to check with 2 and 1 and that is take care of the possibility of accommodating 0 1 2 3 or so. This establishes the correctness of our equation.

In the next step, I am going to determine the complexity, now we will look at the complexity of this algorithm. Now, before I do that I want to make a comment about this step, if you know this, so all we require here is one multiplication, namely A times P. Subsequently, we will have to multiply 2 to the power 2 P or 2 power 1 P or just P notice that these can be computed by simply shifting by 2 bit, 1 bit or n1. So, what we really require is one multiplication and a few summation and summations are linear in time.

So, what we are involving here is one multiplication of size of two numbers of each of size n bits. Now, notice that A cannot be more than n bits unless P is 2 power n minus 1 and that is the special case, one can always ignore that and then these are 2 n bits numbers, so multiplication of 2 n bits numbers may be denoted by M n. So, let M n is time to compute multiplication of 2 n bit integers, now let us get back to this, this is a recursive call.

(Refer Slide Time: 40:18)



Let $M(n) \equiv$ Time to Compute multiplication of two n-bit integers

$$T(n) = T(n/2) + M(n/2) + 2M(n) + kn$$
$$\le T(n/2) + k'M(n)$$
$$= k'(M(n) + M(n/2) + M(n/4) + \cdots)$$
$$\le k'(M(n) + \tfrac{1}{2}M(n) + (\tfrac{1}{2})^2 M(n) + \cdots)$$
$$= k'M(n)[1 + \tfrac{1}{2} + \tfrac{1}{2^2} + \cdots]$$
$$\le 2k'M(n)$$

So $T(n) = O(M(n))$

$$M(n/2) \le \tfrac{1}{2} M(n)$$
$$M(1) = O(1)$$

Sup $n \ne$ Power of 2
then $\exists\, n' = 2^r$
& $n \le n' < 2n$

So $P_{new} \rightarrow 2^{n'-n} \cdot P_{old}$

$\langle p_1 p_2 \cdots p_n 0 0 \cdots 0 \rangle$

$$T(\ ) = O(M(n'))$$
$$\le O(M(2n))$$
$$O(4M(n))$$

$O(M(n))$

So, the time complexity of reciprocation of n bit number is T of n by 2, that would be this step here, this is true here, it is just shifting to linear in number of bits this one though. So, this is a multiplication of 2 n by 2 bit numbers to get C square and then 2 n bit numbers because this is an n bit and this would be n bit number. So, this required M of M by 2 plus M of n, this is linear in number of bits, so and just we said requires one multiplication of n bit, so will make it two and then a few term some constant k having linear term in this. This term can be observed by even notice that M n is at least linear and the other part says that M n by 2 is less than equal to 1 by 2 M n.

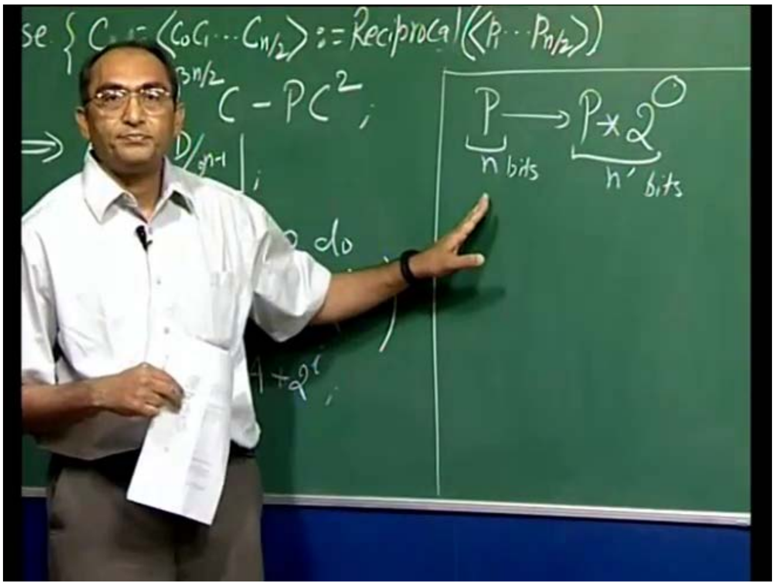
It may be actually worst than linear in general. So, we know that M n by 2 cannot be more than 1 by 2 n, so we have let us say 1 by 2 M n to replace this 2 M n here and may be some k M n here. So, put together this is plus k prime M n for some constant k then I can actually expand this and that would be k prime n by 2 n by 4. N of 1 is 1 order one, this we can now replace by M n plus 1 by 2 M n plus 1 by 2 square M n that is k prime M n the geometric series 1, 1 by 2, 1 by 2 square, this is 1 by 1 minus 1 by 2 which is 2. Therefore, this is less than equal to 2 k prime M n, notice that k prime is a constant.

So, what we have established is P n is of the order of the same time, that it takes to multiply to integers of n that was our objective. Now, we need to address what happens when n is not a power of A, what happens is, but in general n need not be power of 2 then, how do we actually use such an argument to justify this result? Well suppose n is not a power of 2 then there exist n prime which is 2 power some r and n is less than equal to n prime. Well, since this should not be equal and will be less than 2 it is always there is a power of 2 between any numbers, n is doubled, so we can replace P by 2 power n prime minus n times old P, so P new is P old times.

If we have effectively taken our P 1 P 2 P n and put 0's here, so that this becomes the number of bits. Here our powers this is P 1 hence, we can take this number P new and compute the result and what you will find is that the time complexity of the procedure using P new will be of the order of M of n prime. I have the n prime bits which is less than equal to order M into M. Notice that we have this inequality which is fine the 2, we know that multiplication is in the worst case quadratic, so this is still order 4 M of n which is still order m n.

So, it is perfectly justified to use this number because we still can argue that this is nothing but order M n, one last problem about this algorithm. Now, here what we have D 1 is we want to compute and most significant bit of the reciprocal of a given number P. So, what happens if I am interested in computing more accuracy, I want to get say n prime number of most significant bit of actual reciprocal of P, well then this take really points out that if you want higher accuracy in the reciprocal.

(Refer Slide Time: 48:17)



Then you replace P by P times 2 power something make it a larger number. So, instead of n, we have an n prime bits and it has to be power of 2. You go there and then call this procedure, so if this was n bit, now this is some n prime bit and the reciprocal will give me n prime bits of accuracy. Therefore, this procedure does not stop us from getting higher accuracy. So, in the next class we are going to look at how square relates to the computational square relates to the complexity of reciprocal and multiplication.