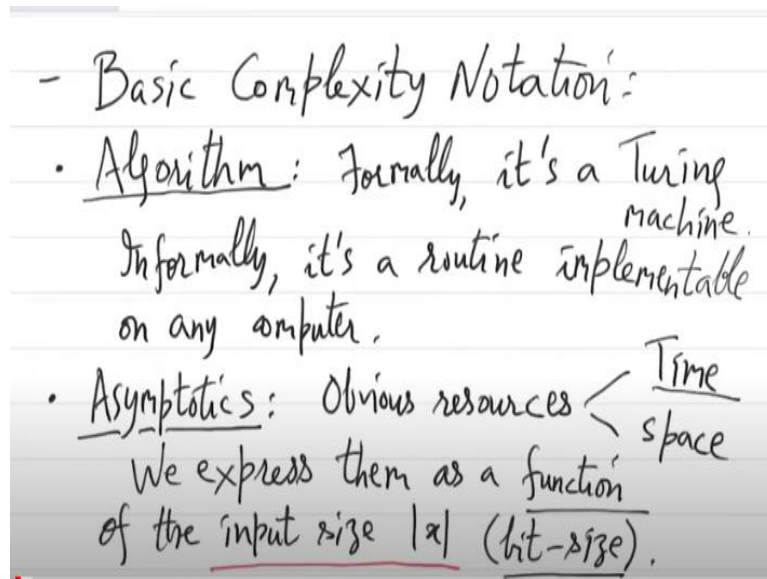


Computational Number Theory and Algebra
Prof. Nitin Saxena
Department of Computer Science and Engineering
Indian Institute of Technology-Kanpur

Lecture - 02
Background

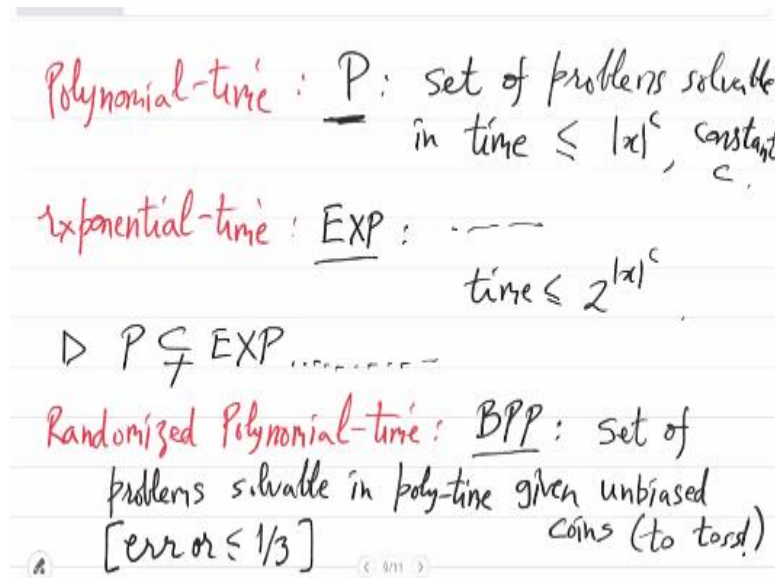
Okay, so any questions from last class? So we did an overview of the course topics and then we started some basic notations that we will be using.

(Refer Slide Time: 00:31)



So basic complexity notation is these keywords algorithm, asymptotics. Whereas algorithm is something which runs on a Turing machine or it is a Turing machine asymptotics is to do with how many resources the Turing machine used. And most importantly, it has to be expressed as a function of the input size and the size is in terms of bits, number of bits.

(Refer Slide Time: 01:07)

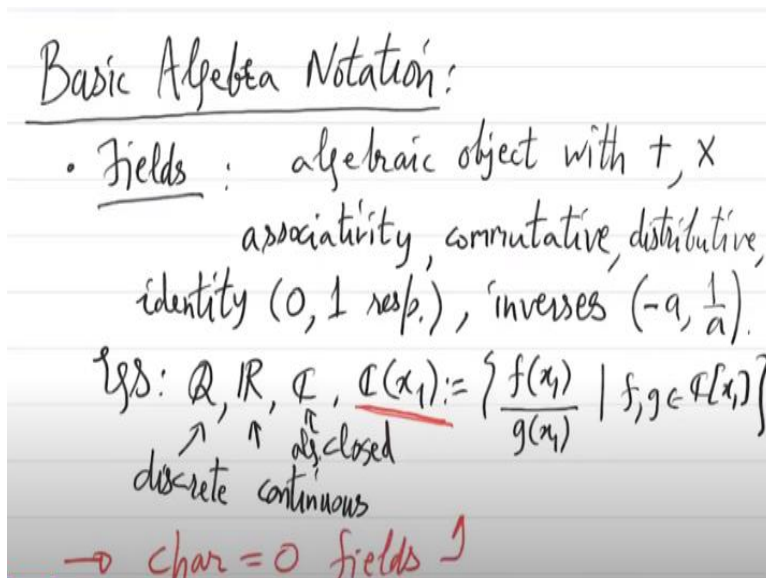


Then there are these complexity classes P , a set of polynomial the deterministic polynomial time, exponential time. These classes are known to be different, EXP being bigger and there is this orthogonal complexity class which is BPP . So which is it is not deterministic, it is like the Turing machine is a probabilistic Turing machine. So the answer it gives is could be wrong with some error probability.

And that probability we have taken to be less than equal to one-third. But it can be taken as any constant below half. And then it can be boosted to something very close to zero. So error can be made almost zero, not exactly zero but exponentially close to zero. So many of our algorithms in this course will be randomized actually. So this is why it is an important complexity class.

There are some problems for which randomized algorithms are very easy. And when you want to make it deterministic, either the question is an open question or the algorithm becomes more technical and less practical. So practical algorithms tend to be randomized in this course.

(Refer Slide Time: 02:44)



So coming to algebra notation, we started with fields. So field is a set. And since we want to make it algebraic, we have to introduce operations and properties. So there are two operations addition, multiplication. Actually, you can call these operations by any name, but we use addition and multiplication because you know this from school, so you can relate it to integers.

But you could have used anything here, not just plus and cross. The properties are important. So they should satisfy associativity. So they should be associative, commutative, distributive. There should be an identity and there should be an inverse irrespective of the element. So if all the properties are there, then it is a field. So like $\mathbb{Q}, \mathbb{R}, \mathbb{C}$ or function fields. So this one is a function field.

So next we will continue with, well so the examples you have seen here all these examples are what is the characteristic of these fields, these four examples? 0 right. So these are characteristic 0 fields. But these are clearly not all the fields that we know. There are far more interesting fields or at least equally interesting fields which have prime characteristic, not zero.

(Refer Slide Time: 04:43)

$\rightarrow \text{char} = p \text{ (prime) [Exercise]}$
 - Exercise: $\mathbb{Z}/\langle n \rangle$ is a field $\Leftrightarrow n$ is prime
 $\mathbb{R} \text{ (integers mod } n)$
 - eg. $n=6$, $\cdot 2^{-1} \text{ mod } 6$ undefined.
 $\cdot 2 \times 3 \equiv 0 \text{ mod } 6$
Exercise: 1) Finite field has size = p -power.
 2) \exists unique field of size p^d .
 (denoted by \mathbb{F}_{p^d})

So characteristic of a field what can it be? It can be 0 of course. Yeah, the other option is only prime. So that I leave as an exercise. So this is an exercise that why should every field, why should every field has characteristic either zero or prime. So what is the example of a characteristic p field? Yeah, so \mathbb{F}_p is the integer ring modulo an ideal which is the ideal generated by the number p okay $\mathbb{Z} \text{ mod } p$.

So in fact, do this as an exercise. You must have done it in some form in some earlier course that \mathbb{Z} modulo the ideal n , well I have not defined ideal yet but this again you should know from some earlier course. So ideal is just a subset of a ring or of the ring in this case \mathbb{Z} , which is not only a sub ring. So it has its own addition, multiplication. But also if you multiply any outside element with an ideal element, then you come inside the ideal, okay.

So for example, any multiple of n is again an element in this ideal and so this is an, this set subset is called an ideal. And with ideal you can do what you are doing here which is the mod operation. So integers mod n , okay. So you can basically, instead of looking at all the integers, you only look at integers 0 to $n - 1$ and when you add them or multiply them, you take a remainder by dividing by n .

And this again miraculously gives you a ring. So it is a different ring from integers okay. So this is a way to create new rings and this is a field. When is this a field? Yeah. So the exercises that show that this is a field only when n is a prime number in

which case n is a prime ideal. So what happens when n is composite? What is the main problem that happens?

Yeah, so the problem is that inverse will not exist. So take n to be 6. Then the problem is that 2 inverse mod 6 does not exist, okay. So there is no number x such that $2x$ is equal to 1 and this you can see by going mod 2. So mod 2 on one side you will get 0, on the other side you will get 1, right. So that contradiction means that two inverse does not exist. So that so this is the only problem.

If n is composite then there are these so called zero divisors, okay. So mod 6 2 is a zero divisor because two times three is zero. That is another explanation. And when n is prime then this possibility is not there. So for prime n essentially every integer has an inverse except obviously multiples of n . So this gives you one finite field. What are the other finite fields?

So this field is clearly finite right because its elements are 0 to $n - 1$. Are there other types of finite fields? Fields with finitely many elements, what are they? Yeah, so you basically then you have to define polynomials over $\mathbb{Z} \bmod n$ or $\mathbb{X} \bmod p$. You define polynomials and then you work with irreducible polynomials. You quotient by that. So these things will be discussed in the first assignment.

So let me here just state it as an exercise. So show that any finite field has size equal to p -power. So if the characteristic of a finite field is p then its size has to be a p -power. There is no other option for finite fields. So these in fact exist because you take up irreducible polynomial of degree d with coefficients in \mathbb{F}_p and you look at $\mathbb{F}_p[x]$ modulo that polynomial. So this will give you a field of size p to the d , okay.

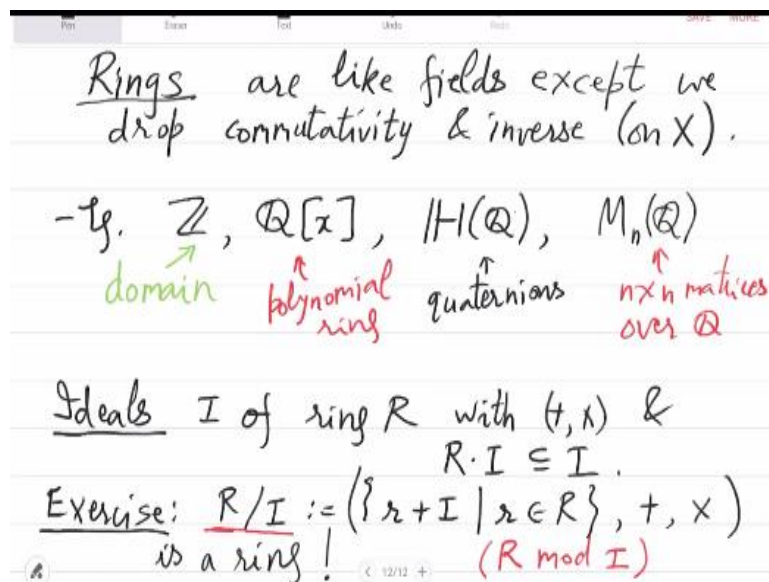
So these exist and also importantly these are unique. So there exists a unique field of size p to the d . And this we will denote, this field we will unique field we will denote by \mathbb{F}_{p^d} . Okay so the work construction of this \mathbb{F}_{p^d} is just $\mathbb{F}_p[x]$ modulo over degree d irreducible.

So the one interesting fact or one interesting consequence of uniqueness is that no matter which degree d irreducible polynomial you take mod p the two fields you will

get will be isomorphic, okay. That is quite a non-trivial fact. So all those things you will see if you work out this exercise, okay. These are very standard properties. Any questions at this point? If you have any fundamental questions then you can ask about these properties.

Otherwise I assume that you will do the homework. So now, you have fields. Did we talk about rings? Yeah we should also talk about ring.

(Refer Slide Time: 12:38)



So ring is like fields but not quite. So some properties are missing except that we will drop commutativity and multiplicative inverse. So these are like fields except we drop commutativity and inverse conditions on both the operations or only one operation, only on multiplication. This is only on the multiplication operation the properties are reduced. So multiplication here is more relaxed.

But addition everything is the same as for fields. Yeah, so this actually covers then a lot of ground. So there are far more rings and complicated rings than there are fields. So example is the ring of integers. It has addition, multiplication and multiplication actually commutativity is there, but inverse is missing sometimes, well actually most of the times.

So in integer what are the elements which have inverses? Only? Yeah so only, so plus minus one are the only units, which means they have inverses. They have an inverse. All the other elements are non-units. Yeah, but these other elements that are non-units

are these zero divisors? Yeah, they are not zero divisors because if you take 2 there is no number x such that $2x$ is equal to 0, right.

So integer is not a bad ring, it is very close to a field. So such rings are called domains. Yeah, so integer is actually \mathbb{Z} is a domain. So that is a special property it has. There are worse rings possible. So another example is when you look at polynomials. So you can look at polynomials over integers or you can look at polynomials over \mathbb{Q} or real or complex. So polynomials always form a ring, okay.

Because always polynomials you can add and you can multiply. And as long as this base ring is good, you will even have commutativity and that you will have commutativity of multiplication also. Inverse is tricky because, in fact even x you would not be able to invert. The symbol x does not have an inverse, right? So it is only the inverse which is a problem otherwise polynomial rings.

So this object is called a polynomial ring. And there are fancier rings like this, the ring of quaternions. So what is this? How many elements do you have here in quaternions? Eight? No. So there are these basis elements are few, but number of elements is infinite. So quaternions are this there is 1 there is i there is j there is k and then i^2 j^2 k^2 each of them is equal to -1 .

So essentially you have three kinds of square root of -1 in this, right. Usually you have only two kinds of square root of -1 . Here you have far more. Here you have around 6 okay. And then i times j is k and so all these properties are there. And then you can take combinations of one i j k . So point being that it is a very bad ring. So it does not have commutativity and it does not have inverse, okay.

And you can make it worse by going to matrices. So matrices are the worst. So if you look at n by n matrices over let us say \mathbb{Q} then all the properties are gone. So two n by n matrices you can add. An addition will have all the properties, but when you multiply the multiplication first of all is non commutative. And usually made and many matrices do not have inverse, right. There are singular matrices.

And in some sense, this matrix algebra or matrix ring covers all other rings, okay. Any other ring you can essentially write down as a matrix algebra. So this is the worse that it gets, at least in the examples that you will see in the course. And yeah, so rings have ideals. So ideal I of a ring R is a sub ring. It is a subset which has full ring structure inherited from R . But it has one, sorry.

It does not, yeah it may have but then it will trivialize. But it has addition, multiplication properties. It is closed under addition, multiplication. It may or may not have 1. But it always has 0. Yeah, so addition is well behaved. Multiplication has issues, but it inherits almost all the properties of the original ring R . In addition, it has this strange extra property with addition, multiplication.

And the strange property that if you multiply any element in R with any element in I , okay then you are inside I right. So this is a strange property. The only motivation for this is when you studied integers modulo n . So this there you are looking at multiples of n and multiples of n that subset has this property. That if you multiply it with any integer then you still get a multiple of n okay.

So just that property has been abstracted out and it has been made crazily general. So any such set now we are calling it an ideal at the level of general rings. The usefulness of this is it is not apparent but you can show it as an exercise that you can now define ring elements modulo I . So it basically simulates division and remainders. So that is an exercise. That $R \bmod I$ make sense and it is defined as abstractly it has these elements $r + I$ for every element r in R in the ring.

And there is addition, operation and multiplication operation okay. So as a set it has these abstract elements. You can think of this as one element per ring element R . But there is also a $+ I$ remember. So it is not R truthfully but some approximation of R okay. So it actually is a it is hashing the whole ring into fewer elements just like integers mod n is hashed hashes \mathbb{Z} an infinite domain into only n elements.

So this again hashes the ring into fewer elements and it has its own addition, multiplication operation. So when you add R_1, R_2 so $R_1 + I$ element and $R_2 + I$ element their sum is the element $R_1 + R_2 + I$ okay that is the definition of addition.

And what is the product? $R_1 + I$ times $R_2 + I$? It is $R_1 R_2 + I$. And that is so to meet that definition meaningful you need this property $R.I$ is contained in I .

Okay, so you do this as an exercise. So $R \bmod I$ is a ring. Okay, this is not at all obvious. This needs exactly the way we have defined an ideal, okay. This is the motivation for the definition of ideal because we want this new ring structure. So we will read this as $R \bmod I$, okay because this is really what we wanted. We wanted to generalize integers modulo a number.

So we can do that at the level of rings okay. So last thing I need is groups. In fact, at this point let me also talk about morphism.

(Refer Slide Time: 24:50)

• Morphisms : homomorphism, isomorphism, automorphism, epimorphism, monomorphism, endomorphism.

$$\phi: (R_1, +, \times) \rightarrow (R_2, +, \times)$$

$$a \mapsto \phi(a)$$

$$b \mapsto \phi(b)$$

$$a+b \mapsto \phi(a) + \phi(b)$$

$$a \times b \mapsto \phi(a) \times \phi(b)$$

So once you have defined an algebraic object you can look at two instances of algebraic objects and you may want to compare them whether they are similar or if they are not similar. Are they approximately similar? And if not, then how far are they, right. So you want to classify algebraic objects once you have defined an algebraic object like you want to classify or compare two fields or two rings or two groups.

So for that you need the concept of morphisms. So examples are homomorphisms, isomorphism, is there anything else? There is automorphism. So let me throw all these keywords and it is your job to read the Wikipedia page what they mean okay. So the

most fundamental is homomorphism. Homomorphism is just a map from one ring to another ring R_1 to R_2 such that it should preserve addition and multiplication.

So ϕ so let us say ϕ is a homomorphism from the ring R_1 to ring R_2 . So maybe let me write that in notation. So ϕ is a homomorphism from a ring R_1 with its operations to another ring R_2 with possibly different very different operations. So at the level of sets, it will be a map from set R_1 to set R_2 and at the level of ring it has to now preserve the structure. So which means that if you have so it sends a to $\phi(a)$ and b to $\phi(b)$.

Now where should it send $a + b$? Exactly. So $a + b$ should not be left free. Since you want structure preservation it should be sent to $\phi(a) + \phi(b)$, okay. This is the only point. So this for plus and this for multiplication, right. So this should be true for all a, b . And from this basic axiom, you can deduce many things. So for example, where should 0 be mapped to? So we have not written it here.

But from this axiom, you can deduce that 0 will be mapped to 0 , and 1 will be mapped to 1 . 1 might be mapped to 0 , you are saying. **“Professor - student conversation starts”** We have to mention that 1 maps to 1 explicitly because inverse property does not exist, if 1 exists. **“Professor - student conversation ends”**. If 1 exists if 1 is in R_1 . Yeah. Yes.

So if no, if 1 is in R_1 , even then it is possible that maybe 1 is being mapped to 0 . But then it would be a trivial ring. Yeah, so 1 you have to be careful with but at least you can deduce that 0 will be mapped to 0 in a homomorphism, okay. So you can deduce many properties. **“Professor - student conversation starts”** Sir, also we have to specify that R_2 has to be a integral domain, not a **“Professor - student conversation ends”**.

No homomorphism does not care about any property at all. So just two rings. So given two rings, you can talk about homomorphism and homomorphism may not exist. Well, I mean there is always this homomorphism that sends everything to 0 . So that is a trivial homomorphism. So trivial homomorphism always exists. But other interesting homomorphisms may not exist, okay.

And then you would know that the two rings are very different. Also if there is a homomorphism what can you roughly say about the sizes of the rings? Which ring is smaller which is bigger. Exactly, so well then I have to talk about image also. Yeah. Well okay. So if as long as R_2 is the image of ϕ , which means everything in R_2 is hit by ϕ , as long as that is true, you can see that R_1 is somewhat larger, right?

It may also be equal in size, but in general it will be bigger and R_2 will be smaller, right. So you are really comparing the various properties, including the size of the rings when you talk about homomorphism. Isomorphism, what is an isomorphism? Yeah, so now once you have defined homomorphism, you can talk about its many, many variants.

So isomorphism will be, a ϕ will be called isomorphism if at the level of sets basically it is a bijection, okay. So every element in R_2 is hit and it is hit in a unique way. There should not be two elements in R_1 , which are mapped to the same element in R_2 . And every element in R_2 should be covered, okay. So it should be a bijection at the level of set, and then it is an isomorphism.

So if ϕ is an isomorphism, then essentially you it is fair to say that R_1, R_2 are the same rings, okay. They are not two different rings, but it is actually the same structure. You have just changed the alphabet that is all. So isomorphism finishes the comparison. There is nothing left to say. What is an automorphism? Right R_1, R_2 are same and ϕ is an isomorphism.

So it is an automorphism. Auto reflects to itself. Then there are fancier names like epimorphism. What is an epimorphism? Yes. So if the map is objective, then it is then you attach a p . And what is a monomorphism? If the map is injective okay, then you can call it a monomorphism. And finally and what is an endomorphism? Endomorphism is a homomorphism from R_1 to itself, okay.

So endo again refers to itself, but in a weak way. So it is not a bijection to itself. Yeah, so most important is homomorphism and then everything else is just a variant. Some properties are added or lost, fine. So when we said that finite fields are unique

of a given size, we meant that any two finite fields of the same size R_1, R_2 if you discover there will also be an isomorphism, okay.

So the only flexibility you have is to change the alphabet. The algebraic structure will remain unchanged, it is unique. That is the meaning of unique in this in the context of algebraic objects. Okay. And finally, we will define groups.

(Refer Slide Time: 34:01)

Groups are objects with a single operation \times (& natural properties):
 (abelian?)
 eg. $(\mathbb{F}, +)$, $(\mathbb{F}^* := \mathbb{F} \setminus \{0\}, \times)$,
 $(GL_n(\mathbb{F}), \times)$,
 $\mathbb{F} \in \text{field}$. \mathbb{R} invertible matrices $n \times n$

Exercise: (\mathbb{F}_p^*, \times) is cyclic group.

- $(G, *)$ is cyclic if $\exists g \in G$ s.t. $\{g, g^{-1}\}$ together generate G using $*$. \uparrow generator =

So groups are algebraic objects which are even more relaxed. There is only one operation, multiplication. So multiplication, basically your set should be closed under multiplication. So any two elements you can multiply and get an element within the set. There should be the identity element which is 1 in this case and there should be inverses, okay.

So those three things closure, unity or identity and inverse and associativity also yes. So four properties. So I would say that the only extra thing here is inverse okay. So it is like a ring without addition but then you want inverse to be present for every element. Then you call it a group. What if inverse is absent? What is the object called? Right. So if so there is also an object name if inverse is missing it is called a monoid.

But we would not care about it in this course. Okay, this course is really about mainly about rings. Even groups we will not study directly in this course. So all computations will be actually ring computations. So examples are field with plus. We cannot say

field with multiplication because zero is non invertible right. So but field with plus is a group.

And but if you remove zero, which is then called F^* . So F^* is $F - 0$. Then we can look at multiplication. So F^* with multiplication this is a group. Every element has inverse associativity, closure, and obviously there is 1. So F is a group, F^* is a group, different operations. And since both of them are commutative examples, these we call abelian groups, okay. So that is a special type of group.

What about matrices? Sorry. Yeah, so matrices with addition they form a group. But if you want matrices with the multiplication, then you have then you are only allowed invertible matrices. So that object is called GL_n . So what does GL_n stand for? Yeah, so GL stands for general linear group of n by n matrices, entries coming from the field F , okay. So since these are all invertible matrices, it is a group under multiplication.

So yeah, maybe I should mention that these are general linear means that we are only looking at invertible matrices. So it is a group almost by definition. F is a field, okay. Okay, so let me leave this with an exercise, which is a very useful property in this course. What can you say about finite field F_p to the d^* with multiplication operation? Yeah, so this is a group as mentioned before.

But the beauty is that it has a single generator, okay. So there is an element G says that G, G^2, G^3, \dots this set is the whole field, is the whole set. It is the whole field minus the zero element. So this is a cyclic group. Okay, this is an interesting property with a relatively elementary proof. So go through the proof of this. In particular, it is already mysterious for $d = 1$. So F_p^* is a cyclic group, right.

So you are looking at integers modulo a prime. And what we are saying is that there is a single number G such that its powers will produce 1 to $p - 1 \pmod p$ right, which already is unexpected. So that is a very special property. So in general cyclic, we call a group cyclic if there is an element G in G such that g, g^{-1} generates everything using star.

So we are talking about g inverse also because just another way of saying that g raised to numbers, both positive and negative, are allowed. And by using these positive and negative exponents you can produce the whole set G . It is important when G is an infinite set. Otherwise you could have only talked about G . Any questions? So this is a lot of terms and definitions, but this soon will become our language.

So all the problems and algorithms will be phrased in terms of this notation. So if you are not familiar with this, then it will be very hard to develop all that machinery which we will develop over this in this course. Okay, so try to get familiar with this. G is called a generator of G . So this is called a generator, okay.

(Refer Slide Time: 42:58)

Part Exercise Text Links Notes Share More

- Ex. 1: $(\mathbb{Z}, +)$ is cyclic group with generator 1.
 $\langle -1, 1 \rangle =$

- Ex. 2: $(\mathbb{Z}/n\mathbb{Z}, +)$ " "
 $\langle 1 \rangle =$

Exercises: 1) Any ∞ cyclic group is isomorphic to $(\mathbb{Z}, +)$.
 2) Any size- n cyclic group " " to $(\mathbb{Z}/n\mathbb{Z}, +)$.

< 15/15 >

So for example, integers with the addition operator it is a group. Is it cyclic or not? Yes. So what is the generator of \mathbb{Z} plus. Right. So if you just use 1, then you see that you can only generate the positive numbers. So you also need -1. But with 1 and -1 you can generate everything, right. So this is cyclic. So a cyclic group with generator 1. So identity element here is the generator.

And the way it generates is -1, 1 and then use the addition operation again and again. Okay another example. So integers mod n . What about this group? Well first of all it is a group right because every element has an inverse. a has inverse $-a$. Is it cyclic? So again this is generated by 1. So here you just use 1 and you do not even need -1 because you just keep adding 1 you will reach n and that is 0.

After that you will come back to, you loop back to 1, right. So that is the modern arithmetic. Okay. So simple exercises. So what can you say about a cyclic group that is infinite? Can you classify an infinite cyclic group? Exactly. So that is the first thing you prove that any infinite cyclic group is isomorphic to \mathbb{Z} . Okay this is quite easy to show. You just have to properly understand the definition of cyclic group.

So you are saying that, that there is an element G such that G , G square on the positive side and G to the 0, G to the -1 on the negative side. This is your whole set. So from that you set up an isomorphism with integers, okay. You basically bring down the exponent. So formalize that as a proof. Show the isomorphism. So remember that was in terms of multiplication.

I think we had defined group in terms of multiplication only. Yes. So in multiplication, it is G , G square, G cube. But when you convert it into addition, then it will come down to integers. So just do that as an exercise. It will give you some familiarity. And what about size n cyclic group? Right. Isomorphic to what? Permutation? No. Just the second example. $\mathbb{Z} \bmod n$. So any size n cyclic group is isomorphic to $\mathbb{Z} \bmod n$.

So I may say $n\mathbb{Z}$ or I may say the ideal n it is the same thing. Okay these are basically just multiples of n . So the ring \mathbb{Z} you are quotienting out or you are modding out by the ideal n , multiples of n . And then you are only looking at the additive structure. And this is really the only cyclic group of size n , okay. So there are these simple characterizations of cyclic groups. Any questions? Okay.

(Refer Slide Time: 48:51)

Asymptotics

- Let $f, g: \mathbb{N} \rightarrow \mathbb{R}^+$. The comparisons are:

$$\begin{array}{lcl} f(n) = O(g(n)) & , & g(n) = \Omega(f(n)) \\ \text{upper bound} & & \text{lower bound} \\ f = o(g) & , & g = \omega(f) \end{array}$$

$$f = \tilde{O}(g) : f = \Theta(g) \\ f = O(g \cdot (\log g)^c) \\ \text{for constant } c.$$

So now let me move to asymptotics before we start doing something more interesting. So asymptotics is something that will always come at the beginning and the end of our lectures. So our theorem statements will always be in terms of how fast are we solving a problem, okay. So how fast will always be quantified in terms of asymptotics.

And once we have given an algorithm, in the end we will always show that we will do an analysis basically time complexity analysis of each of the steps okay. So there also we will do in terms of asymptotics. We cannot really go into the precise number of steps. So that will both be painful and boring, okay. So to avoid that we will purely work we will assume that certain simple things only take constant time.

And we will not say what that constant is okay. So that is essentially why we need asymptotics. So this is basically a way to compare two functions, okay. So let f, g be two functions positive valued. So the comparisons are so you can say that f is at most g , right. So that you will say by writing f equal to big O of g . Is there anyone who has not seen this notation, big O of g .

Raise your hand if you have not seen it. Okay, so I would not define it. So this is just saying that so remember these are functions right? So f , there is this implicit n sitting here, $f(n)$ $g(n)$. So when we say that the function f is big O of g , what is meant is that there is an absolute constant let us say c such that $f(n)$ is less than equal to c times $g(n)$ for increasing values of n . So in the beginning things may be bad.

So f and g may be fluctuating a lot, but after a point, let us say n greater than equal to n_0 , n_0 is again an absolute constant. After this breakpoint $f(n)$ will always be smaller than c times $g(n)$ or equivalently the ratio of f and g will never exceed c okay, after the breakpoint. So that is what is meant which is actually the same as saying that $g(n)$ is what? So f is at most g . So then g is at least f .

So this you say by big omega, right. These two things are the same. So big O and big omega and then you swap the functions. So the first is called an upper bound and the second is called a lower bound, okay. This term we may also use sometimes and for non-CA students it may be confusing. So this is what we mean precisely. Okay, and then you may want to make it stricter.

So you can say that f is equal to small o of g . What is the meaning of that? Yeah, so now if you look at the ratio f by g it actually tends to zero okay. I mean think of f as the square root g okay. So whatever is g f is significantly smaller than that. So let us say square root g . So when you will, as n keeps growing this f over g tends to zero in the limit.

And similarly, you can say, I mean it is equivalent to saying that g is little omega f okay? So these are the strict versions of upper bound and lower bound. So if you have both upper and lower bounds on f , then you can use theta. So f equal to theta g . So theta g basically is combines the two things f equal to big O of g and f equal to big omega of g , okay. So g gives you both sides of f , okay.

So as once n crosses a breaking point, then f is at least $c_1 g$ and f is at most $c_2 g$ for absolute constants c_1, c_2 . But this is not used so much. You do not use theta so much because our problem in life is that we never know about lower bounds, okay. So we only give algorithms which gives you an upper bound. The algorithms never tell you what is a lower bound.

So lower bound may be something very small and you may be spending a lot of time in the upper bound. So the gap is always very large. So theta is very sparingly used. Big O is used most often. In this course, we will also be using soft O . So what is soft

O? You soften it by adding a tilde. And what is the meaning of this? Yeah, so maybe this needs a definition.

So this is defined as f is equal to big O of g times log of g raised to an absolute constant, okay. So if in our time complexity analysis we get a function g times let us say log g to the 100 then we say that log g is a very small function, even this thing raised to 100 does not matter much. So let us just suppress it and call it O tilde g. So soft O allows these arbitrarily many but still only constantly many log factors

Okay, so let us see some actual problems and complexity, which you should already know. So arithmetic in easy arithmetic in Q.

(Refer Slide Time: 57:07)

1. Easy arithmetic in \mathbb{Z}

1) $a \pm b$ in $O(\log|a| + \log|b|)$ - bit operations. (time)

2) $a \times b$ in $O(\log|a| \cdot \log|b|)$ - time.

3) q & r s.t. $a = \underline{q} \cdot b + \underline{r}$ ($0 \leq r < b$)
in $O(\log|q| \cdot \log|b|)$ - time.

Let us say for example suppose you are given a number a in bits and you are given a number b in bits and you want to add or subtract them, okay. So what is the time complex? What is the algorithm you know and what is the time complexity of that algorithm in terms of asymptotics? Yes, so a is an integer or a yeah maybe not rational but integer. Let us simplify this in fact.

So a is an integer with a sign. So we will forget about the sign. Think of a is a positive integer. So how many bits will it have? It will have log a many bits like so log a base 2 many bits. And then when you want to add in the school way, you will put a like this and b like this and then these are the bits. And then you will do the addition. So this is essentially a single scan. So you are just scanning from right to left.

So the time complexity of this algorithm is $\text{big } O(\log a + \log b)$ time. So to be precise, actually these are the bit operations. And when we say time, we really mean this, okay. So at the level of Turing machine which only understands bits, is your time and the big O just hides the constants because related to your algorithm, there is a Turing machine and there is some overhead.

But all those things are absolute constants. We do not want to go into those details, those implementation details. So we just abstract it out by big O . I mean in terms of programming, you can think of it as so you have this first array for a and you have second array for b . And then you might need some temporary variables etc., to add things and then keep something for carry forward.

And then for, again for addition you have some other temporary variables. So all those all that overhead is there which we are not mentioning here, but it is only constant okay. So this is why it is fair to say that the time complexity is $\text{big } O(\log a + \log b)$. So in the same spirit, what about $a \times b$? So the same school algorithm when you do for multiplication, then you have to do it for each of the bits of b .

Yeah, so for each bit of b , you have to scan a , right? So it will be $\log b$ times $\log a$ and we will call it time. So addition, multiplication and finally division. So you have to do long division. So you want to find q and r , quotient and remainder such that a is equal to q times b plus r ; r is smaller than b . So how do you do this long division in how much time? You are given a and b in bits again and you want to find q and r .

It is very similar to multiplication. Its time complexity is exactly the same as multiplying q with b , right because you are, bit by bit you are discovering the quotient and then you that you use to multiply it with the b and then do a subtraction with the dividend, right. So the time complexity is $\log q$ times $\log b$. Is it okay? So the long division works this way.

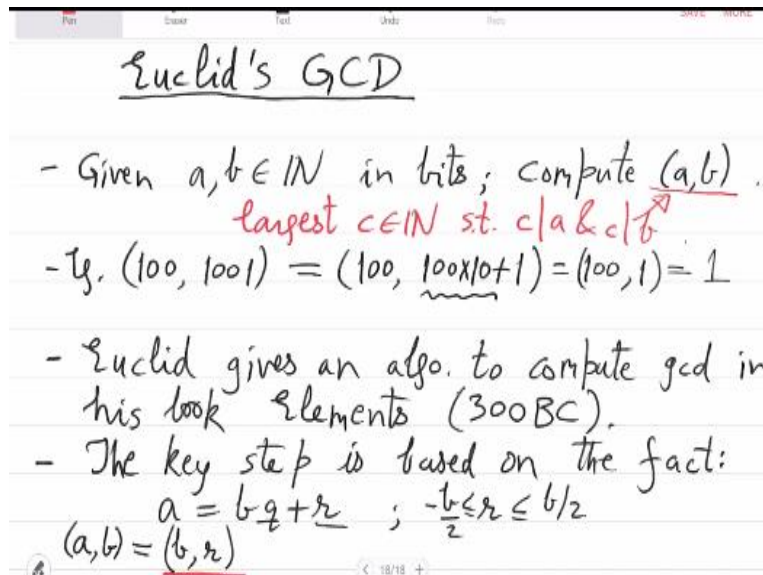
So you first find q_0 , whether it is 0 or 1. You will first find that and you will multiply it with b and then you will subtract from a , right. Now I am making a mistake. Is this greater than b ? So is the question that the time complexity of subtraction we are

missing. So the thing is that what is how big is this box? This box is only as big as b right? So it does not matter how big a is.

You are only looking at the first few bits of a , $\log b$ many bits of a , okay. So this subtraction which is which we are doing this subtraction has, it only depends on actually $\log b$. Does not depend on a . Okay, this is an important point. So this we will very soon see that this $\log q$ times $\log b$ complexity we need to compute GCD okay. So it is better that you convince yourself that for long division the complexity is \log quotient times \log divisor.

The a and r ; r anyways is very small; a will not appear in the complexity. I mean obviously, implicitly it appears because there is q and there is b . So if a , as a grows q also grows. So implicitly obviously it appears but only in this form. Is that clear? Okay. So those are the simple things that you should familiarize yourself with. So let us now look at the first non-trivial algorithm, which is Euclidean GCD.

(Refer Slide Time: 1:06:05)



Euclid's GCD

- Given $a, b \in \mathbb{N}$ in bits; compute (a, b) .
largest $c \in \mathbb{N}$ st. $c|a$ & $c|b$
- Eg. $(100, 1001) = (100, 100 \times 10 + 1) = (100, 1) = 1$
- Euclid gives an algo. to compute gcd in his book *Elements* (300 BC).
- The key step is based on the fact:

$$a = bq + r \quad ; \quad -\frac{b}{2} \leq r \leq \frac{b}{2}$$

$$(a, b) = (b, r)$$

So this again, I am sure you have seen because we teach it, at least in ESC 101. You have to write the recursive program. But here we will do a threadbare analysis of it. Okay, so exactly what is the complexity and later on, we will also improve the complexity by using more advanced means. So the question is, as you know, you are given a and b . Let us say non negative integers in bits. And you want to compute the GCD.

So I will just write in short bracket a comma b. That will mean GCD of a comma b which is defined as largest c that divides both. So c divides a and c divides b, okay. So this is the notation we will use for division. So you want the largest number that divides both a and b. That is called GCD. And this is the notation for it.

So the way this definition is written, it seems that it is necessary to factorize a right, because so if you recall the prime factorization theorem, any number like a can be factorized into primes, right, and c also can be factorized into primes. So one trivial way to compute c is you just identify these primes and their frequency and that you can do by factorizing a, right.

So you identify the primes appearing in a and check whether they divide b. And then you find also a frequency, the multiplicity of p required such that, that p power divides both a and b. And then you multiply and get c, right. But this is a horrible algorithm because even if a is 300, 400 bits this search for primes, prime factors is highly expensive, okay. So fortunately for very different reasons an extremely fast algorithm was found which does not need factorization of a.

Okay, so without factoring a, you will be able to find a factor of a that is shared with b. So a simple example of this is 100, 1001. What is the GCD? Yes why is it 1? What is the proof? You are basically saying that these two numbers are coprime. So when the GCD is 1, we call numbers to be coprime okay. Otherwise we say that they share a factor. So 100 has many factors.

So you look at the factors and you check that they do not divide 1001. Or you just do this calculation, right. So you see that 1001 when divided by 100 gives remainder 1 and whatever this part is this is not important for GCD conclusion. So you drop this part and then you are left with 100 with 1. So obviously the GCD is 1, right? So this is the starting point of Euclid's GCD algorithm.

It will not try to factor 100, it will just try to divide b by a or a by b. So this is a very old algorithm. It might even be the first algorithm that human beings found and which is really fast compared to brute force. So Euclid gives an algorithm to compute GCD

in his book called elements in Greek okay. This is from 300 BC. And so the algorithm is just a sequence of long division and it will actually give you more than just GCD.

Okay, it has some surprising consequences. So let me just say that the key step is that if you have divided a by b to get q and r . Well usually you assume that the remainder is between 0 and $b - 1$. But just for fun say the remainder is even smaller than b by 2 . So let us say that the remainder is smaller than b by 2 and at least $-b$ by 2 , right.

This also can be achieved, because if the remainder comes out to be more than b by 2 , then you just subtract b and you will go in the negative side. So this will be actually very important in the analysis of the algorithm or the analysis will become easier to see with this assumption. So GCD of a comma b can now be simplified using r , okay. So what is that simplification?

Yeah, so note that since bq is a multiple of b and you want to take GCD with b , it does not matter what q is. So you drop that and you get r comma b . That is the key point, okay. That a suppose was very large and b was small. So now the GCD computation has to be has been turned over to a much smaller instance, right. Now b is small and r is smaller. And then you repeat this.

So this will very simply give you a very fast algorithm and an easy analysis. Okay, stop.