

of length 1 they do not have inverses. So, let us try and and construct inverses or somehow introduce inverses. And to do this there is sort of one key idea that we use ok which is the following. Instead of looking at words in this alphabet S , we look at words in an augmented alphabet ok.

So, what does that mean? I am going to enlarge the set S to make it somewhat bigger. There is a, b and I will put in two more elements which I will call a', b' these will eventually perform the role of the inverses ok.

But for now think of them as sort of two more alphabets, and we look at the set of all words in this 4 element alphabet. So, that is well that is still a very large set, its now got all words in using a, b, a', b' , its got the empty word, its still got an associative multiplication and that associative multiplication still does not have inverses ok. So, we have sort of just replaced the original set words of S by a much larger set in some sets ok. But here comes the key step. So, we consider this, so that is step 1 . Step 2 is really the key to everything that we are about to do and this is a construction which will keep coming up in many places, which is the following; we impose the following relations ok.

So, we impose the following relations what does that mean? Well first let me tell you what the relations are. So, I want look at the word aa' , this is a word of length 2 in a words of S hat. This I want to say that this word should be the same ok. So, I will just put the tilde there to say the same as or equivalent to the empty word ok.

So, I want to similarly make $a'a$ equivalent to the empty word ok. So, I want to somehow impose these four relations ok. So, notice that at the moment a you know the things that I have written on the left hand side are all words of lengths 2, and of course, on the right hand side we have the empty word of length 0 and of course, these are not equal definitely right now.

But what we want to do, is to somehow say that we want to think of them as being equivalent ok and not just this, we impose these relations and further what else do we want to do, we also want to say that we can also replace ok . So, here sort of, I mean think of these two steps right now as just being some sort of a heuristic idea what it means for a' and b' to perform the roles of the inverses of a and b ?

So, at the moment do not think of them you know, do not be too worried about the lack of rigor, we will make it a little more precise soon. So, heuristically here sort of what we want to do. We impose these relations and what else do we also want? We also want to do the following . Whenever we we have a word say we have some long word in this alphabet a, b, a', b' and somewhere in the middle of this word if I find any one of these 4 words of length 2.

So, for example, something something something aa' followed by a bunch of other letters, I want to be able to take this word and sort of remove the aa' block from the middle ok. So, I want to think of it as the same word in which the aa' block is removed from the middle. And then so I just take the words to the left of this block and concatenate it with the the words to the right of this block.

For example so, let us just do this by example . So, I will go to the next page. So, here is an example . So, if I have for example, $abb'a'ab$. Now look at this word . When I scan it I notice that you know that is sort of this bb' which occurs one after the other. There is also another $a'a$ which occurs one after the other ok.

So, in fact, there are two reductions here that I would like to be able to do. So, if a' is to be the inverse of a then we would want this word to be the same as or to be equivalent

$$(eg) \quad abb'a'ab \sim abb'b \sim ab$$

$$abb'a' \sim aa' \sim \epsilon$$

What we want to do is to define an equivalence relation
 \sim on words (\hat{S})

Define: Given $w_1, w_2 \in \text{Words}(\hat{S})$, we say
 $w_1 \sim w_2$ if w_2 can be obtained from w_1 by a
 sequence of basic rewriting rules



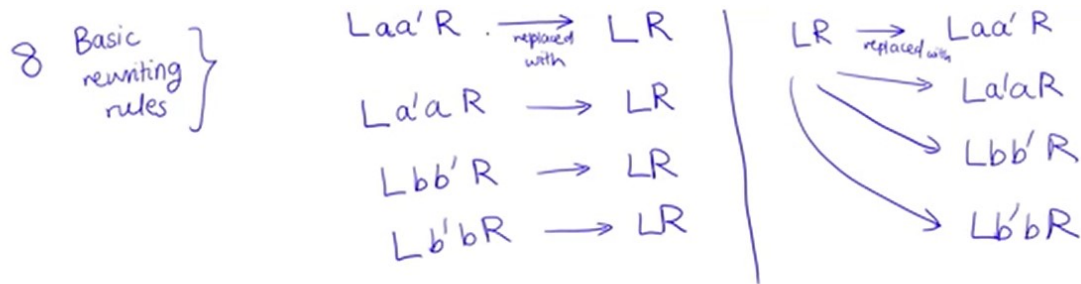
to the following word. So, let me make the second $a'a$ into just the identity, so it just becomes this.

And now, I can look at this pair or in fact, if I choose I can also look at the other pair, let us do the one in the middle if I replace that with the empty word, then I want this to just collapse to the word ab ok. So, this is one sort of reduction that I would want which is what it means to say that a' and a are inverses of each other b' and b are inverses of each other. So, I mean to take a maybe um slightly different example, I can take $abb'a'$ for example. And here again I can I would want to think of this word as being equivalent to the word aa' , and similarly the word aa' is equivalent to the empty word ok.

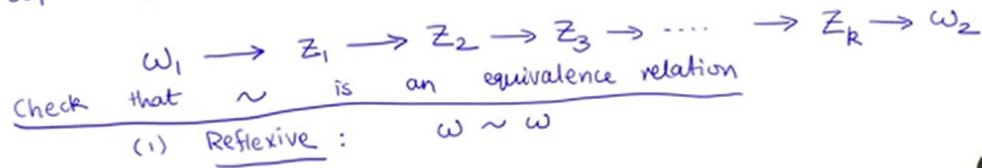
So, what exactly are we getting at at this point? So, this is really captured precisely by the notion of an equivalence relation ok. So, if you sort of think a little bit about what we are doing? Here is what it is. So, what we want to do really is to define an equivalence relation ok. So, which I will denote by tilde on the set of all words in the alphabet \hat{S} ok. So, here is really what we are trying to do. Now the question is how to rigorously define such an equivalence relation. So, we have already seen some of the requirements. We have those 4 important identifications. We would want these five words $aa'a'abb'b'b$ and the empty word to be equivalent to each other ok. So, that is the primary requirement. And then the second property here is that whenever any of these occurs as a sub word of a longer word then inside that longer word you can replace the sub word by any one of these other 4 sub words ok. So, that is sort of what we want to do.

So, let us try and define this equivalence relation in a formal manner ok. So, we say define. So, here is the definition:

Definition: Given two words w_1 and w_2 in \hat{S} , we say or we define w_1 is equivalent to w_2 , if w_2 can be obtained from w_1 by a sequence of what I will call the basic rewriting rules.



$$\omega_1 \sim \omega_2$$



So, let us see what is what is meant by this? So, we have already seen an example of this just above . So, what is a basic rewriting rule ok, those are just ones that we have talked about. So, the basic rewriting rules are the following. So, let me say let us ok. So, there are 8 basic rewriting rules. So, what are the rewriting rules, it says suppose I have a word . So, if there is a word in which aa' occurs ok and there is something to the to the right. So, this is maybe its easier to to call these two guys something, so let us call this whatever occurs to the left of aa' as the left sub word L .

So, that is again a word in the alphabet and what occurs to the right . So, let us just delete the dots and call this as L , so some word and what occurs to the right of a prime is R . So, given a word of this form some word L followed by aa' followed by some word R . I am allowed to rewrite. So, this can be replaced with the word $L a a' R = LR$ ok. In other words I am I am just going to delete the aa' from the middle. Similarly if I have a word L followed by $L a' a R = LR$. Again I am allowed to replace that word which is the word LR ok. So, let us write the other two .

So, when you get, so we say that the word LR is obtained from the original word $L a a' R$ by using one of the basic rewriting rules ok, so these these are called the basic rewriting rules. And these are four of them now there are four more which are just the same rules in reverse ok.

So, not only are you allowed to delete an aa' or an $a'a$ from a sub word from a word, you are you are also allowed to introduce that if you want ok. So, here are four rules and the other four are just the reverses of these . So, given two words L and R , I am allowed to introduce an aa' in the middle or I am allowed to introduce ok. So, I am allowed to replace LR with replaced with this with a with a word whose length is two more ok. So, these are the 8 basic rewriting rules that I have.

So, what are the rewriting rules they are a way of obtaining a new word from a given word ok. The first 4 rules give you a new word whose length is two less and the next 4 rules give you a new word whose length is two more ok . Now, using any of these 8 basic rewriting rules in sequence you are allowed to use them again and again one after the other in any order you want ok. Now keep doing this and starting from a word w , if w_1 if you can somehow obtain w_2 ok.

So, going back to our definition we said $w_1 \sim w_2$ what does that mean? It means I start with w_1 ok, I see if one of these 8 rules can be obtained, I mean can be applied . I mean you can always apply one of the rules, because for example, the rules which allow you to add two letters in the middle that can be applied to any word if you wish. So, I take w_1 I apply one of my basic rewriting rules and it gives me some word ok. So, I do not know I should call it something, maybe I get a word z what shall we call it z_1 and I take this word z_1 I apply one other basic rewriting rule maybe it gives me a word z_2 and on z_2 , I apply one of the rules I get z_3 etcetera etcetera . And I keep doing this say at some point again z_k and I finally, apply a rule and it transforms into w_2 ok.

$$w_1 \rightarrow z_1 \rightarrow z_2 \rightarrow z_3 \rightarrow \dots \rightarrow z_k \rightarrow w_2$$

So, if you can do this, if you can find a sequence z_1, z_2, z_k ; such that this chain from $w_1 \rightarrow w_2$ proceeds through successive basic rewriting rules, then you declare that $w_1 \sim w_2$ ok. So, I hope the definition is clear, its slightly involved its not may be not kind of equivalence relation definition which you have seen before , but nevertheless. So, the first order business let us check that this is actually an equivalence relation on the set ok.

So, let us check is this an equivalence relation check that ok. So, remember equivalence relation means. I need to check three things I need to check that the relation is reflexive which means given a word w . I need to check that it is equivalent to itself and that is sort of trivial, because I just do not apply any of the basic rewriting rules right.

I am allowed to apply you know any number of them, but I am also allowed to apply just zero number of basically writing rules. In other words I do not apply any rule, or if you really wish to you know to actually apply a rule you can apply for example, the say the 5th rule and then follow it up by acting the the 1st rule ok. So, you add in the aa' and then you delete the a' if you wish. So, its in fact, a reflexive operation .

Let us show the other two properties symmetry says that . So, this is a second requirement of an equivalence relation is that it should be symmetric. So, if $w_1 \sim w_2$ then I need to show that this means $w_2 \sim w_1$ ok, but again observe this is why we had 8 basic rewriting rules and not just four the first four. The the rules 5 through 8 are just sort of the rules 1 through four in reverse right.

So, if I have say some word w gives me a word x by using one of the basic rewriting rules. So, let me say if for example, I apply rule 1 to go from $w \rightarrow x$, then its clear that I can get $w \rightarrow x$ by just applying rule 5 right. If I can get this by deleting a pair aa' then I can get the other guy by just putting that pair back right.

Similarly, if I get a word $x \rightarrow w$ by applying say rule what was that two then I can go back, I can undo it by using rule 6 etcetera ok. So, its clear that if I construct a chain going from $w_1 \rightarrow w_2$ then by applying the rules in reverse I can just get a chain going from um $x \rightarrow w$ ok . And 3rd property transitivity is again as easy to check ok. So, if I have a chain which connects $w_1 \rightarrow w_2$, chain of rewriting rules and I have a chain of rewriting rules connecting $w_2 \rightarrow w_3$. All I have to do is just you know apply this first set of rules follow it up by the

(2) Symmetry: $\omega_1 \sim \omega_2 \Rightarrow \omega_2 \sim \omega_1$

$\omega \xrightarrow{\text{Rule 1}} \chi$ $\omega \xrightarrow{\text{Rule 2}} \chi$ etc
 $\xleftarrow{\text{Rule 5}}$ $\xleftarrow{\text{Rule 6}}$

(3) Transitive: $\omega_1 \sim \omega_2 \ \& \ \omega_2 \sim \omega_3 \Rightarrow \omega_1 \sim \omega_3$



next set of rules and using that rewriting I can start with w_1 , find the chain which takes me all the way to w_3 ok.

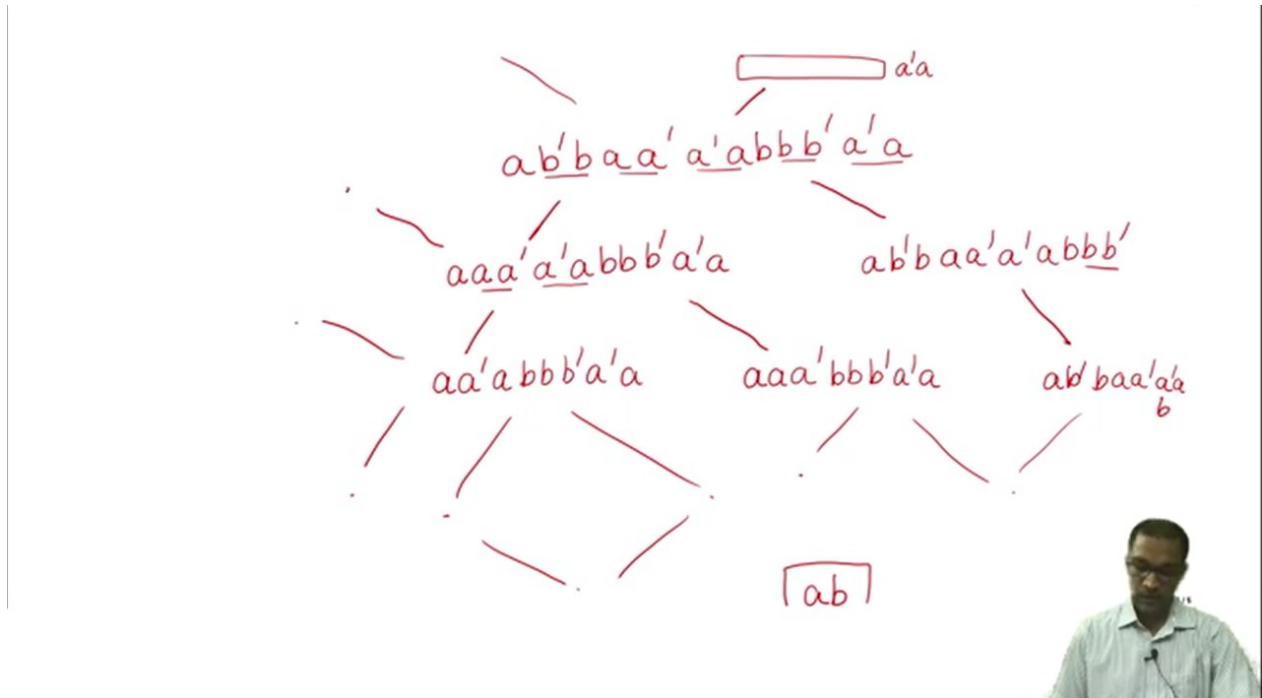
So in fact, this somewhat strange definition defines an equivalence relation on the set words of \hat{S} ok. So, that is the the first thing that we have checked ok. So, now let us maybe just do an example um of of applying the sequence of rewriting rules and so on. So, for instance let me start with a word that has $ab'b$ ok. So, here is a word a with a is b is a' and b' in it and let me see what I can do by way of applying my rewriting rules .

So, I already see that there is $ab'b$ right there. So, there are many other pairs which can be collapsed. For example, there is abb' here, there is an $a'a$ there, there is an $a'a$ here, there is an aa' here and so on.

So, I can of course, apply my rewriting rules um one at a time. So, when I apply my rule this guy becomes; for example, let me collapse the first pair, so this then becomes $aaa'a'abb'b'a'a$. Or if instead of the first pair I choose to collapse the last pair $a'a$, so then that is like $ab'baa'a'abb'b'$ ok.

Now, so as you can see you can keep going, so I apply, let us see what should we apply next suppose I collapse this aa' then it becomes $aa'abb'b'a'a$. Then again here I have other choices I could have collapsed let us see maybe the the next pair $a'a$. So, that will give me $aaa'bb$ and so on. So, this again maybe I collapse the last $bb'a$ and there is ab at the end ok and so on.

So, you can see that you will sort of form a structure like this and of course, its its its an infinite um set, the the words you can get from a fixed word by using the rewriting rules, because I am always allowed to add um you know I can always insert an aa' or $a'a$ wherever I want. For example I could take this this original word whatever it is, and maybe just take $a'a$ at the end ok, I just add one extra a' here ok and so on. So, you can imagine that its like this. So, some of these will will eventually become the same meaning this is and so on ok.



And in this case as you can probably see from what we we underlined already, if you did the all possible collapsings to the maximum extent I have $b'b$ is gone, aa' is gone, $a'a$ is gone, bb' is gone, this is gone. So, finally, after all this you will see that you will sort of get the word ab ok.

So, that is going to be a word which will which will eventually arise when you keep doing this. And in some sense that is the smallest words the the the smallest the maybe you should call it the the the word of smallest length or sometimes its called the reduced word and so on. But here is a nice exercise um you know for you to play with, just try to see what more um you know try to write down a bunch of other words that you can construct which can all be obtained one from the other by a sequence of rewriting rules ok.

Now, our definition of the equivalence relation what it says is that for example, all the words that you see now on the page are all in fact, equivalent to one another ok why is that because no matter where I am. So, for example, if I have this word here, this word here is actually equivalent to this word here ok. These two words are equivalent why because what is the definition say I should be able to go from one to the other by applying one of my basic rewriting rules .


So, here is what I do I first go up like this ok. So, going from here to here involves insertion of a pair aa' and then from there I reach this word which again was obtained by insertion and then I go down like this, where this word is obtained from the word on top by deletion of some pair and then I reach this point ok. So, no matter where I am I can always start there and reach any other point in this collection of words, either by using insertions or deletions of pairs aa' ok. So, this entire collection is in fact, equivalent. Everything here is equivalent to everything else and this what we mean by an equivalence class is more or less this this idea here .

\sim An equiv relation partitions the set into a collection of disjoint equivalence classes. \downarrow words(\hat{S})

$[w] :=$ the equivalence class of w
 $= \{ x \in \text{words}(\hat{S}) \mid w \sim x \} \subseteq \text{words}(\hat{S})$

$G :=$ The set of equivalence classes for \sim
 $= \{ [w] : w \in \text{words}(\hat{S}) \}$

Note $\underbrace{[w]} = \underbrace{[x]} \quad \forall x \in [w]$



So, recall that when I have an equivalence relation. So, an equivalence relation always divides or partitions the set, the underlying set, partitions the set into a collection of disjoint what are called equivalence classes. So, this is disjoint equivalence classes ok. So, it partitions the set in this case the set itself is nothing, but the set of all words ok, where what is an equivalence class.

So, here is if you recall the definition. So, suppose I pick a word w I will denote by this square bracket $[w]$ the equivalence class. So, let us use this to denote the equivalence class to which w belongs. So, this is the equivalence class of $[w]$ which by definition is just nothing, but the set of all words:

$$[w] = \{ x \in \text{word}(\hat{S}) \mid w \sim x \}$$

I mean collect every word which is equivalent to w that collection is called the equivalence class of w . And of course, what this is is a subset, this is a subset of the set of all words in \hat{S} ok.

And now, here is the important definition for S . So, let us call this set G is just a set of all equivalence classes, so G is the set of equivalence classes ok for this equivalence relation tilde that we have just defined. Just take the collection of all equivalence classes. In other words the elements of G are themselves subsets of words of \hat{S} ok.

Now, look at the collection of equivalence classes and observe that, so one way of writing its to say its the set of all w , as w varies over set of all words in \hat{S} , but note that you know if I take an equivalence class of $[w]$. So, w sometimes you know when we denote an equivalence class in this manner what we are really doing is, picking one representative w from that equivalence class, but note that the following is true if I picked another element x from the same equivalence class. So, I pick an element x from the equivalence class of $[w]$, then the equivalence class.

Proposition: G has a well-defined binary operation given by

$$[w_1] \cdot [w_2] := [w_1 * w_2]$$

Proof: Need to show: If $x_1 \in [w_1]$ & $x_2 \in [w_2]$

(i.e.), $[w_1] = [x_1]$, $[w_2] = [x_2]$.

Then: we must show: $[x_1 * x_2] = [w_1 * w_2]$

Given: $x_1 \sim w_1$ & $x_2 \sim w_2$

Need: $x_1 * x_2 \sim w_1 * w_2$



So, for all x in the equivalence class of $[w]$, the equivalence class of x is just the same as the equivalence class of $[w]$ ok. In other words the same equivalence class in this case this you know either of these two sides they represent the same equivalence class, but the elements that you pick to represent it which in this case are w and x can very well be different ok . So, that is something to always keep in mind when one is dealing with equivalence classes ok.

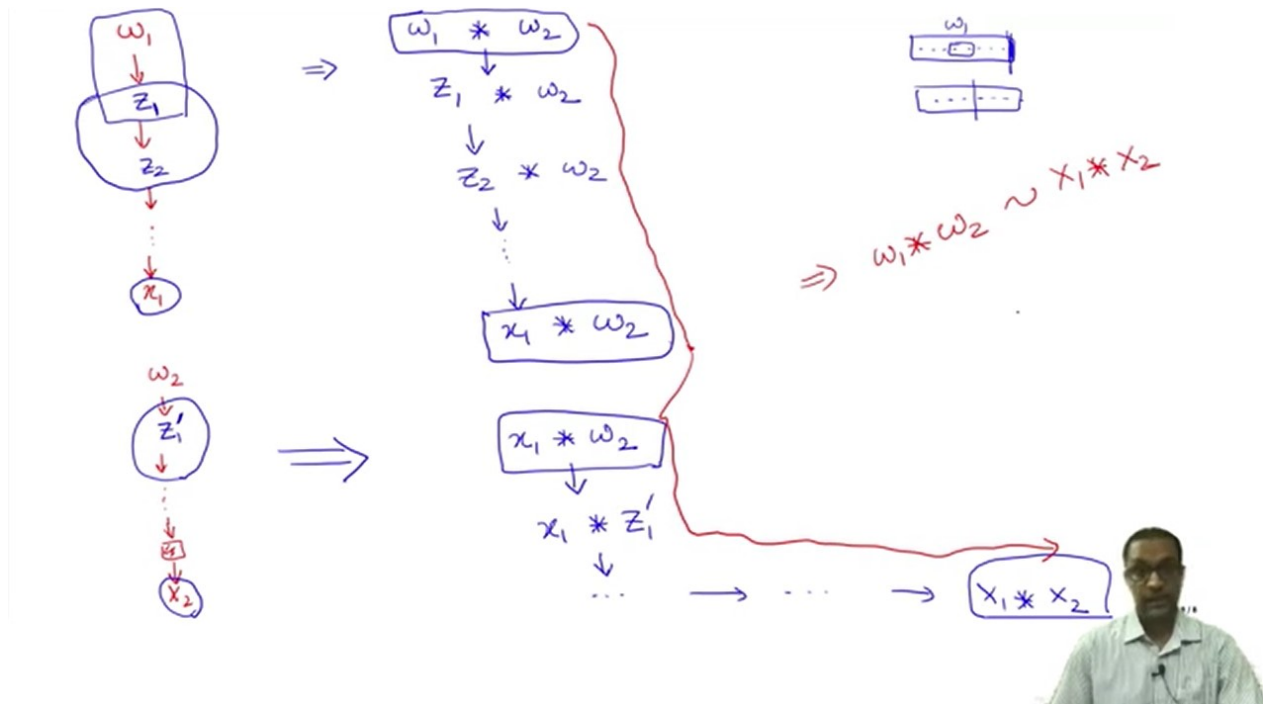
Now, here is the very interesting observation that the way we have defined the the equivalence relation this collection G also has a well defined. So, G has a well defined a binary operation, basically a multiplication operation given by \cdot . So, here is the definition remember elements of G are themselves equivalence classes. So, I take the equivalence classes of class of $[w_1]$ and here is this dot denotes the binary operation the multiplication I am going to define on G . So,

$$[w_1] \cdot [w_2] := [w_1 * w_2]$$

So, observe the the definition itself at the moment makes sense, I take two equivalence classes $[w_1]$ and $[w_2]$, I take two representatives w_1 w_2 from those classes. I I concatenate those two representatives I get a new word $w_1 * w_2$ and I take the equivalence class of that new word ok that is my definition.

Now, the key thing that one needs to prove is really well definedness ok. The key point here is that this definition is well defined and why does one need to worry about that because we are picking representatives from these classes w_1 and w_2 if .

So, its not a priori clear that suppose I changed my representative. So, here is what we will need to show what if I pick different representatives. So, if x_1 is another representative it belongs to this equivalence class ok and I pick a different representative x_2 from the equivalence class $[w_2]$ ok. Then so, what does that mean? I have instead of taking w_1 as my



representative I am thinking of x_1 as my representative of that class and similarly ok. So, I am I am changing my representative then. So, suppose I do this, then I need to show that my right hand side will be the same answer whether I do $[w_1 * w_2]$ and take the concatenation or I take $[x_1 * x_2]$ and and take their concatenation, the answers should be the same. its ok. Then um we need ok we must show that $x_1 * x_2$ the equivalence class is the same answer as what you would get if you took $w_1 * w_2$ ok. So, this is what showing well definedness means ok. So, let us prove this. Again it it comes about from the special way in which the equivalence class or the equivalence relation was defined.

So, observe what are we given we given that x_1 and w_1 are equivalent to each other. So, let us prove this given what is given is the following these two are equivalent and what we need to prove is that the concatenations are equivalent $x_1 * x_2 \sim w_1 * w_2$ ok. So, what is equivalent mean? It means there is a chain starting with w_1 ending at x_1 in which each intermediate step is obtained from the preceding one by an application of a basic rewriting rule ok.

So, let us write that down. So, step 1; I started w_1 ok and I use my basic rewriting rule, I get some word , I use again one of the other rules it becomes some other word, etcetera etcetera till I finally, am able to reach x_1 ok. This is what it means to say w_1 and x_1 are equivalent to each other ok. Similarly, I have w_2 and x_2 . So, I can start at w_2 , I can follow my rules successively till I reach well ok. So, that is what the definition says.

Now, given this we need to show that if it start with $w_1 * w_2$, I can reach $x_1 * x_2$ by means of my basic rewriting rules ok. Now, the way we do this is just via this simple observation that look at this this initial chain, starting from w_1 and going to x_1 . What does the basic rewriting rule do ? The basic rewriting rule has the following form. So, for instance if this is my word w_1 , I look through the letters in w_1 , I either find a successive pair of the form aa' etcetera which I delete or I take my word w_1 . I look through the letters, I pick some


Theorem : G is a group

Pf: a) \cdot is associative

$$\begin{aligned}
 ([\omega_1] \cdot [\omega_2]) \cdot [\omega_3] &= [\omega_1 * \omega_2] \cdot [\omega_3] \\
 &= [(\omega_1 * \omega_2) * \omega_3] \\
 &= [\omega_1 * (\omega_2 * \omega_3)] \\
 &= [\omega_1] \cdot ([\omega_2] \cdot [\omega_3])
 \end{aligned}$$

b) Identity
 $e = []$
 $[\omega] \cdot [] = [\omega *] = [\omega] = [] \cdot [\omega]$

c) Inverses exist!
 $[a] \cdot [\dots] = []$



position and there I insert a pair aa' ok. So, the transformation that I perform has this very specific form. You either collapse or you sort of expand at a certain location in the word.

So, if I go from let us look at the very first step of the chain I go from w_1 to whatever is the next word in the sequence by means of some basic rewriting rule then here is what it means, it means that look at w_1 and the next guy. Now, to w_1 on the right hand side, I can let me try doing this. Let me concatenate w_1 on the right hand side with the word w_2 ok and the next step in the chain I will do the same thing to it. I concatenate w_2 to that word.

Now, if this second guy can be obtained from the first one by some basic rewriting rule, then it follows that $w_1 * w_2$ will lead to that word $x_1 * w_2$ ok. I can take the second guy and I do w_2 to it. So, maybe I should give this give this a name. So, this intermediate step suppose, I call it z_1 that is my word, then here is what I mean that if from w_1 to z_1 I can go by means of a basic rewriting rule. I can also go from $w_1 * w_2 \rightarrow z_1 * w_2$ by the same rule really ok, because I only need to apply that rule to the w_1 portion of my word now and so on.

So, the next step again, because I can go from z_1 to the next fellow z_2 by means of some rule, it means I can also go from $z_1 * w_2 \rightarrow z_2 * w_2$ by using the same rule, but only applying into the z_1 portion and so on ok. So, keep doing this till you reach the end. So, the end here is x_1 , but of course, remember I have starred every one of them with the w_2 ok.

So, I have started at $w_1 * w_2$ which is what I need to start with ok. So, we have managed to go from $w_1 * w_2$ which is what we wanted, but we have only reached $x_1 * w_2$ ok, but that is already a very good start, because now we do the same thing with the with the other sequence. We know that we can start at w_2 and reach x_1 . So, now let us do the same thing here this implies. Now, I will sort of concatenate on the left of w_2 . So, starting at at w_2 , I can reach the next step whatever this this word is. So, let us call this z'_1 now, what I do is to concatenate x_1 on the left of w_2 and, because I can go from $w_2 \rightarrow z'_1$ it also implies that

I can go from $x_1 * w_2 \rightarrow x_1 * z'_1$ in the same manner and you know by the same logic as before.

So, I keep applying it to the next um to the next step of the chain and so on . So, I keep doing this till I reach the very last step and the last step is x_2 , but concatenated on the left by x_1 ok. So, what this means is if I start at $x_1 * w_2$ by applying left concatenation to the second chain I can reach $x_1 * x_2$ ok. So, now I just put these put these together. So, what have I managed to do? I have managed to start at $w_1 * w_2$. I am following this chain of rewriting rules. So, at this point this and this are the same now I follow after that this chain of rules and finally, I reach my destination ok. So, what this means is that $w_1 * w_2$ is in fact, equivalent to $x_1 * x_2$ ok as required. So, what this means is that my um on my group G or at the moment my set G the um binary operation that have defined is at least it it makes sense its well defined ok, but we called it G for a reason we are going to make this into a group .

We are going to show that it actually becomes a group under this operation and that is really our our in some sense our main theorem that is going to be the free group. So, G is a group under the binary operation that we just defined ok proof well what all do we need to show we need to show that the binary operation is associative ok. Why is it associative? So, will show that first observe the definition said if I take $[w_1]$ multiplied by $[w_2]$ ok, let us just write out the definition of associativity here. This by definition is $[w_1 * w_2]$ the equivalence class of the concatenation multiplied by the equivalence class of $[w_3]$ ok which by definition again is the equivalence class of $[(w_1 * w_2) * w_3]$ ok.

But observe that the what is inside, the representative of the class that I get here is just the concatenation of these three guys $[w_1 * (w_2 * w_3)]$ and the concatenation operation is of course, associative. So, I can replace this triple with say $[w_1]([w_2][w_3])$ and that by the definition of the multiplication in G will just become this product ok and that is exactly the the verification of associativity ok.

Now, the identity is also easy. So, there is an identity element what should the identity element be? Well, so, let us call it e maybe , the identity element of this group G well I claim its nothing, but you take the equivalence class of the empty word ok. So, that is the the empty word in words of \hat{S} the equivalence class of the empty word by the ways it has lots and lots of words remember right. So, we um looked at for example, $aa'a'a$, those basic rewriting guys they are all. In fact, in the equivalence class of the empty word so this is in fact, the set of all words which if you keep applying rewriting rules will finally, come down to the empty word.

The claim is that that serves as an identity again by definition, because the multiplication just says for any word e , if I multiply it with the equivalence class of the empty word, I just have to concatenate w with the empty word, but that is just w ok and observe the same logic holds in the other order. If I hit the equivalence class of the empty word with w on the right then of course, it gives me w ok. So, in some sense these two properties just follow from the corresponding properties of the $*$ operation. So, no surprises so far, but the reason we were doing all this is, because the $*$ does not admit inverses ok, but the important property here is that this does have inverses ok. In other words it is a group.

So, let us verify this, let us check that given any element of G . I can construct an inverse of that element with respect to this this new operation that we have defined. So, for a start let us let us just look at the the two basic generators a and b , the alphabets um if I just take the single , if I take the equivalence class of the the word a , the question is what is the inverse of this guy.

$$[a] \cdot [a'] = [a * a'] = [aa'] = [e] = e$$

$$[a'] \cdot [a] = e = [b] \cdot [b'] = [b'] \cdot [b]$$


(eg) $w = abbaa$ $[w] \cdot [a'a'b'b'a']$

$\bar{w} =$ read w in reverse // & replace each letter by its dash.

$$[w * a'a'b'b'a']$$

$$[abbaaa'a'b'b'a']$$

$abbaaa'a'b'b'a'$



So, in other words what equivalence class will you take such that you know what should I put here. So, that this product gives me the identity element and remember the identity element just means it is the equivalence class of the empty word ok and observe that well by definition we already know something. So, let us just throw that in let us let us perform a simple computation here. So, observe if I take a and I multiply it with the equivalence class of the element a' , then by definition this is the equivalence class of the concatenation. The word a with a' which again by definition is just aa' the word of length 2, but remember aa' by the rewriting rule is the same as the empty word ok.

So, in other words it says that a equivalence class multiplied by the equivalence class of a' is in fact, the identity element of this group ok and of course, in the other order as well a' multiplied by a would also give you identity, you know you can also check that the same is true of b and b' .

So, this is what I meant when I said in the beginning that you know a prime and b prime will eventually perform the roles of the the inverses ok. That is only we have only constructed inverses for these special one letter words if you wish. Ah, What about a general word? That is that is also very easy. So, let us just do it by example and you will you will see the general picture very quickly. Suppose, I take the word $w = abbaa$ for example, ok. So, the question is what should the inverse? So, maybe I should make it slightly asymmetric. So, let me put two a 's on the end. So, $abbaa$ for example, ok. So, take this word w , I want to know what is the inverse of this, this word ok.

So, I claim here is a simple prescription, you just look at this this string of letters you read them in reverse ok and when you read them in reverse you you just change all the a is to a' and b is to b' . So, here is the prescription the inverse of w so, let me call it \bar{w} for now. This is just read w in reverse and replace any any symbol that you see by its dash ok. So, maybe I should call it replace each letter by its dash ok. So, in other words here, I I apply


$$w = aba'a \quad [w]^{-1} = [a'a'b'a'] \quad \text{check!}$$

←

$$w = x_1 x_2 \dots x_k \quad x_i \in \{a, b, a', b'\}$$

$$[w]^{-1} = [y_k y_{k-1} \dots y_1] \quad \text{where } y_i = \text{"the dash" of } x_i$$

Defn (G, \cdot) is called the free group on a, b .



this prescription, I read this in reverse and I see two a 's from the end. So, I make them dashes I make b 's into dashes. So, here's my claim is that this this new word will serve as the inverse. So, let us check. So, how should we check by definition this product is just w concatenated with this new word $a'a'b'b'a'$, which as we know is $abbaa'a'b'b'a'$. So, here is some some long word, but I claim that if I keep applying my rewriting rules successively I can convert this word into the the empty word ok.

So, let us just do this sort of right there. So, let me write down this this word and I will convert it into the the empty word. So, $abbaa'a'b'b'a'$ ok. So, let us just do it right right there. So, I have this pair a and a' right over here. So, I know that that pair can be erased. I can make it into the identity into the empty word. So, I erase that from a word ok. Now, I look at what is left. Now, again in what is left I see that an a and an a' are paired up. So, I can erase them ok. Now, I look at what is left, I see that this b and this b' are paired up. So, I erase them ok.

What is left the b and this b' are paired up, this a and this a' are paired up ok. So, you see that is exactly how we we constructed the inverse. We just read the word in reverse and for every symbol we just took its its dash ok and the reason for doing this is, because exactly this sort of pair wise cancellation is going to happen ok.

And in fact, this is this was just a particular example, but more generally the word can have dashes as well ok. So, here is the maybe here is another example. So, if I have $aba'a$ for example, then I claim that the inverse of that equivalence class is just read the same thing in reverse, but put dashes. If you see a put it put an a' dash, if you see an a' you convert that to an a . So, that is it is in some sense a' is like an a . So, claim its this element ok and I leave this for you to check ok.

And it is easy to write a general prescription as well, if you wish that if w is a word which looks like $x_1 x_2 \dots x_k$ each $x_i \in \{a, b, a', b'\}$, then the inverse of $[w]$ is just given by the

equivalence class of well read in reverse. So, y_k and I will change all the $x's \rightarrow y's$ where what is y_i is just the of x_i ok. So, this being a bit loose here, but you you know what I mean. Each x_i if it is an a , then y_i is a' , if it is a' then it is an a ok.

So, what have we done we have therefore, proved that this group. So, finally, we have managed to do what we set out to do. We have constructed a group G , with respect to this binary operation. So, G with respect to this binary operation is called so definition, this is called the free group on the two symbols a and b ok. So, observe it is it certainly contains in some sense the elements a and b and next time, we will start looking at some of its other properties ok.

For now at least the the motivation for this this terminology, I mean it is not yet fully clear, but we at least said let us start with a and b , let us not put any relations just sort of let a and b be arbitrary symbols. Let them generate look at all possible products and so on and in some sense what we did by this enlargement procedure is to say let us not just take products of $a's$ and $b's$, let us sort of also take their inverses, let us also throw in two formal symbols which are like the inverses of a and b and sort of take all possible products, all words with a in which $a's, b's$ are a^{-1} and b^{-1} occur ok and in some sense that is really what the what the free group is capturing ok. So, more on this next time .