6.041SC Probabilistic Systems Analysis and Applied Probability, Fall 2013
Transcript – Recitation: Communication over a Noisy Channel

Hi. In this problem, we'll be talking about communication across a noisy channel. But before we dive into the problem itself, I wanted to first motivate the context a little bit and talk more about what exactly a communication channel is and what "noise" means. So in our everyday life, we deal with a lot of communication channels, for example, the internet, where we download data and we watch videos online, or even just talking to a friend. And the air could be your communication channel for our voice.

But as you probably have experienced, sometimes these channels have noise, which just means that what the sender was trying to send isn't necessarily exactly what the receiver receives. And so in probability, we try to model these communication channels and noise and try to understand the probability behind it. And so now, let's go into the problem itself.

In this problem, we're dealing with a pretty simple communication channel. It's just a binary channel, which means that what we're sending is just one bit at a time. And here, a bit just means either 0 or 1-- so essentially, the simplest case of information that you could send. But sometimes when you send a 0, the receiver actually receives a 1 instead, or vice versa.

And that is where noise comes in. So here in this problem, we actually have a probabilistic model of this channel and the noise that hits the channel. What we're trying to send is either 0 or a 1. And what we know is that on the receiving end, a 0 can either be received when a 0 is sent, or a 1 can be received when a 0 is sent.

And when a 1 is sent, we could also have noise that corrupts it. And you get a 0 instead. Or you can have a 1 being successfully transmitted.

And the problem actually tells us what the probabilities here are. So we know that if a 0 is sent, then with probability 1 minus epsilon naught, a 0 is received. And with the remaining probability, it actually gets corrupted and turned into a 1. And similarly, if a 1 is sent, then with probability 1 minus epsilon 1, the 1 is correctly transmitted. And with the remaining probability epsilon 1, it's turned into a 0 instead.

And the last bit of information is that we know that with the probability p, any random bit is actually is 0 that is being sent. And with probability 1 minus p, we're actually trying to send a 1. So that is the basic setup for the problem.

And the first part that the problem asks us to find, what is the probability of a successful transmission when you have just any arbitrary bit that's being sent. So what we can do here is, use this tree that we've already drawn and identify what are the cases, the outcomes where a bit is actually successfully transmitted. So if a 0 is sent and a 0 is received, then that corresponds to a successful transmission.

Similarly, if a 1 is sent and a 1 is received, that also corresponds to a successful transmission. And then we can calculate what these probabilities are, because we just calculate the probabilities along the branches. And so here implicitly, what we're doing is invoking the multiplication rule.

So we can calculate the probabilities of these two individual outcomes and their disjoint outcomes. So we can actually just sum the two probabilities to find the answer. So the probability here is p times 1 minus epsilon naught-- that's the probability of a 0 being successfully transmitted-- plus 1 minus p times 1 minus epsilon, 1, which is the probability that a 1 is successfully transmitted.

And so what we've done here is actually just looked at this kind of diagram, this tree to find the answer. What we also could have done is been a little bit more methodical maybe and actually apply the law of total probability, which is really what we're trying to do here. So you can see that this actually corresponds to-- the p corresponds to the probability of 0 being sent. And 1 minus epsilon naught is the probability of success, given that a 0 is sent.

And this second term is analogous. It's the probability that a 1 was sent times the probability that you have a success, given that a 1 was sent. And this is just an example of applying the law of total probability, where we partitioned into the two cases of a 0 being sent and a 1 being sent and calculated the probabilities for each of those two cases and added those up.

So that's kind of a review of the multiplication rule and law of total probability. So now, let's move on to part B. Part B is asking, what is the probability that a particular sequence of bits, not just a single one, but a sequence of four bits in a row is successfully transmitted? And the sequence that we're looking for is, 1, 0, 1, 1. So this is how I'll denote this event.

1, 0, 1, 1 gets successfully transmitted into 1, 0, 1, 1. Now, instead of dealing with single bits in isolation, we have a sequence of four bits. But we can really just break this out into the four individual bits and look at those one by one.

So in order to transmit successfully 1, 0, 1, 1, that whole sequence, we first need to transmit a 1 successfully, then a 0 successfully, then another 1 successfully, and then finally, the last 1 successfully. So really, this is the same as the intersection of four different smaller events, a 1 being successfully transmitted and a 0 being successfully transmitted and two more 1's being successfully transmitted.

So why are we able to do this, first of all? We are using an important assumption that we make in the problem that each transmission of an individual bit has the same probabilistic structure so that no matter which bit you're talking about, they all have the same [? error ?] probability, the same probabilities of being either successfully transmitted or having noise corrupt it.

So because of that, it doesn't really matter which particular 1 or 0 we're talking about. And now, we'll make one more step, and we'll invoke independence, which is the third topic here. And the other important assumption here we're making is that every single bit is independent from any other bit. So the fact that this one was successfully transmitted has no impact on the probability of the 0 being successfully transmitted or not.

And so because of that, we can now break this down into a product of four probabilities. So this becomes the probability of 1 transmitted into a 1 times the probability of 0 transmitted into a 0, 1 to a 1, and 1 to 1. And that simplifies things, because we know what each one of these are. The probability of 1 being successful transmitted into a 1, we know that's just 1 minus epsilon 1.

And similarly, probability of 0 being transmitted into a 0 is 1 minus epsilon naught. So our final answer then is just-- well, we have three of these and one of these. So the answer is going to be 1 minus epsilon naught times 1 minus epsilon 1 to the third power.

Now, let's move on go on to part C, which adds another wrinkle to the problem. So now, maybe we're not satisfied with the success rate of our current channel. And we want to improve it somehow. And one way of doing this is to add some redundancy. So instead of just sending a single 0 and hoping that it gets successfully transmitted, instead what we can do is, send three 0's in a row to represent a single 0 and hope that because we've added some redundancy, we can somehow improve our error rates.

So in particular what we're going to do is, for a 0, when we want to send a 0, which I'll put in quotes here, what we're actually going to send is a sequence of three 0s. And what's going to happen is, this sequence of three 0s, each one of these bits is going to go through the same channel. So the 0, 0, 0 can stay and get transmitted successfully as a 0, 0, 0.

Or maybe the last 0 gets turned into a 1, or the second 0 gets turned into a 1, or we can have any one of these eight possible outcomes on the receiving end. And then similarly, for a 1, when we want to send a 1, what we'll actually send is a sequence of three 1's, three bits. And just like above, this 1, 1, 1, due to the noise in the channel, it can get turned into any one of these eight sequences on the receiving end.

So what we're going to do now is, instead of sending just a single 0, we'll send three 0s, and instead of sending a 1, we'll send three 1s. But now, the question is, this is what you'll get on the receiving end. How do you interpret-- 0, 0, 0, maybe intuitively you'll say that's obviously a 0.

But what if you get something like 0, 1, 0 or 1, 0, 1, when there's both 0s and 1s in the received message? What are you going to do? So one obvious thing to do is to take a majority rule. So because there's three of them, if there's two or more 0s, we'll say that what was meant to be sent was actually a 0. And if there's two or more 1s, then we'll interpret that as a 1 being sent.

So in this case, let's look at the case of 0. The majority rule here would say that, well, if 0, 0, 0 was sent, then the majority is 0s. And similarly, in these two cases, 0, 0, 1 or 0, 1, 0, the majority is also 0s. And then finally, in this last case, 1, 0, 0, you get a majority of 0s.

So in these four received messages, we'll interpret that as a 0 have being set. So part C is asking, given this majority rule and this redundancy, what is the probability that a 0 is correctly transmitted? Well, to answer that, we've already identified these are the four outcomes, where a 0 would be correctly transmitted. So to find the answer to this question, all we have to do is find the probability that a sequence of 0, 0, 0 gets turned into one of these four sequences.

So let's do that. What is the probability that a 0, 0, 0 gets turned into a 0, 0, 0? Well, that means that all three of these 0s had no errors. So we would have the answer being 1 minus epsilon 0 cubed, because all three of these bits had to have been successfully transmitted.

Now, let's consider the other ones. For example, what's the probability that a 0, 0, 0 gets turned into a 0, 0, 1? Well, in this case, we need two successful transmissions of 0s, plus one transmission of 0 that had an error.

So that is going to be 1 minus epsilon naught squared for the two successful transmissions of 0, times epsilon naught for the single one that was wrong. And if you think about it, that was only for this case-- 0, 0, 1. But the case where 0, 1, 0 and 1, 0, 0 are the same, because for all three of these, you have two successful transmissions of 0, plus one that was corrupted with noise.

And so it turns out that all three of those probabilities are going to be the same. So this is our final answer for this part. Now, let's move on to part D. Part D is asking now a type of inference problem. And we'll talk more about inference later on in this course.

The purpose of this problem-- what it's asking is, suppose you received a 1, 0, 1. That's the sequence of three messages, three bits that you received. Given that you received a 1, 0, 1, what's the probability that 0 was actually the thing that was being sent.

So if you look at this, you'll look at it and say, this looks like something where we can apply Bayes' rule. So that's the fourth thing that we're covering in this problem. And if you apply Bayes' rule, what you'll get is, this is equal to the probability of 0 times the probability of 1, 0, 1 being received, given that 0 was what was sent, divided by the probability that 1, 0, 1 is received.

So we have this basic structure. And we also know that we can use the law of total probability again on this denominator. So we know that the probability that 1, 0, 1 is received is equal to the probability of 0 being sent times probability of 1, 0, 1 being received, given that 0 was sent, plus the probability that 1 was sent times the probability that 1, 0, 1 is received, given that 1 is sent.

And as you'll notice in applications of Bayes' rule, usually what you'll have is a numerator is then repeated as one of the terms in the denominator, because it's just an application of total probability. So if you put these pieces together, really, what we need is just these four terms. Once we have those four terms, we can just plug them into this equation, and we'll have our answer.

So let's figure out what those four terms are. The probability of 0 being sent-- well, we said that earlier. Probability of 0 being sent is just p. And the probability of 1 being sent is 1 minus p. That's just from the model that we're given in the problem. Now, let's figure out this part.

What is the probability of a 1, 0, 1 being received, given that 0 was sent? So if 0 was sent, then we know that what really was sent was 0, 0, 0, that sequence of three bits. And now, what's the probability that 0, 0, 0 got turned into 1, 0, 1?

Wall, in this case, what we have is one successful transmission of a 0, plus two failed transmission of a 0. So that one successful transmission of a 0, that probability is 1 minus epsilon naught. And now, we have two failed transmissions of a 0.

So we have to multiply that by epsilon naught squared. And now, for the last piece, what's the probability of receiving the 1, 0, 1, given that 1 was actually sent? Well, in that case, if a 1 was sent, what was really sent was a sequence of three 1s. And now, we want the probability that a 1, 1, 1 got turned into a 1, 0, 1.

In this case, we have two successful transmissions of the 1 with one failed transmission. So the two successful transmissions will have 1 minus epsilon 1 squared. And then the one failed transmission will give us an extra term of epsilon 1. So just for completeness, let's actually write out what this final answer would be.

So probability of 0 is p. Probability of 1, 0, 1, given 0 is, we calculated that as 1 minus epsilon naught times epsilon naught squared. The same term appears again in the denominator.

Plus the other term is, probability of 1 times the probability of 1, 0, 1, given 1. So that is 1 minus epsilon squared times epsilon 1. So that is our final answer. And it's really just a application of Bayes' rule.

So this was a nice problem, because it represents a real world phenomenon that happens. And we can see that you can apply a pretty simple probabilistic model to it and still be able to answer some interesting questions. And there are other extensions that you can ask also. For example, we've talked about adding redundancy by tripling the number of bits, but tripling the number of bits also reduces the throughputs, because instead of sending one, you have to send three bits just to send one.

So if there's a cost of that, at what point does the benefit of having lower ever outweigh the cost of having to send more things? And so that's a question that you can answer with some more tools in probability. So we hope you enjoyed this problem. And we'll see you again next time.