Hi. My name is GW Zhang, and I’m here to tell the technologies that are used to automate these jobs.

First let’s look at a little bit of the history. Before the computers were invented, there are actually person who computes complicated math expressions. But quickly people found it rather complicated so they turned to automation for help. As things get more complicated, people then seeks for methodology in order to systematically simplify these problems.

And the core of automation I would like to mention consists of 2 parts: Abstraction and Algorithms.

Why are they important, or, why abstraction? Well, there are 3 parts.

First, to make problems clear. For example, you have the following problem to solve. <*Reads the problem*>. To solve the problem, we may regard each vertex as a state on the original shore. For example, PGWC means on the original shore there are Person, Goat, Wolf, and Cabbage. We may let the person take the goat to the other shore, so the state would transform to Wolf and Cabbage. Thus, the solution of the problem is to find the shortest path here, which can be easily implemented.

Second, proper abstractions made it possible to maintain complicated systems. It is usually called ‘Black-Box Abstraction’. For example, we can construct many circuits with wires to denote logic relationship like AND NOT and so on. When we come to build more complex systems, instead of connecting the wires randomly, we may regard these basic components as a whole, or rather, black boxed, to make full use of the power of combinations. More precisely, we may notice the following four parts while making a black box abstraction.

Third, it’s friendly to machine to represent data in this way. Like many automation machines, the state of an automation machine is fairly limited. For example, in a computer, only 0 and 1 is ‘physically recognizable’ by the circuits behind. So, after layers after layers of abstraction, we finally made it possible to make computer somehow understand what we have said.

But how should we build the abstraction? Well, that’s the second part, the algorithms come to help.

Algorithm may make ‘abstractions’ dynamic. For example, we have made the abstraction that the problem is ‘find the page of word *s* in *Oxford Advanced Learners’ Dictionary*’, there’s an algorithm that make it work properly. <Reads the algorithm1>. However, there’s another algorithm. <Reads Algorithm2>. We may find that Algorithm 2 is more efficient than Algorithm 1. So, after constructing an algorithm, it’s better to show the algorithm is correct and is somehow optimized.

Finally, I’d like to talk about the application in real world said previously. In an ETC system as Lingjun Sun mentioned above, a huge database system is critical, which involves problems like data racing, concurrency problems, and efficiency bottlenecks. Also, signals received by receiver is also a mean of abstraction.

In the automation production process in the factory, simulation process is critical, which involves abstraction. And the stabilization of the body of the robots requires PID algorithms. The same pattern occurs exactly in dish washing machines.

Let’s look at something more exciting. We may automate the verify process of mathematical proofs. We may have many rules, and design proper algorithms to calculate them. <Demo>

With the development of the size of the data, web applications that give access to databases  
on distant host computers seen gradually plays a significant role in data providing, which is the essential part of processing data automatically.

To sum up, the fundamental, core technologies used in automation is abstraction and algorithms. They give us infinite power and possibility to do any possible things – maybe with a single mouse click, proper abstraction and powerful algorithms work together to send the data from distant databases to your computer.