1. I think O-notation became popular because it provides an easy way to understand how an algorithm scales with input size, which is often the most critical factor in practice, and the O-notation does not require you to have machine related knowledge so that you can understand the size. It allows developers and researchers to spend more time on figuring out which way or algorithm they can use to make the time or memory more efficient. Also, it is easy to compare by using O-notation.

2. When two algorithms have the same time complexity, we can compare them in practical ways. Testing with real input shows actual speed, and constant factors or memory use can make one faster. It also helps to check worst-case and average-case performance, try real data from your application, and consider how simple each is to implement. Profiling can then reveal which one runs better in practice.