In this tutorial we will give an overview of the main principles that underly the operation of both symbolic and on-the-fly model checking tools. Symbolic model checkers are traditionally applied mostly to hardware verification problems, while on-the-fly model checkers, such as the Bell Labs model checker SPIN, primarily target software verification. We’ll review how these different domains of application have prompted the adoption of different types of model checking algorithms.

We’ll also explain and compare the main specification formalisms that are in use for expressing logical correctness requirements (LTL, CTL, etc.). The feasibility of the application of a model checking tool to a real-world problem often depends critically on the types of abstraction that are made. Indeed, abstraction may be the most powerful tool that the user of a model checking tool can apply. In the tutorial we will try to give examples of both good and bad uses of abstraction.