**Submitted by:** Nabeel Khawar

**Submitted to:** Sir Adnan Riaz

**Roll no:** 201245

**Section:** BSSE – VI – A

---

## Model Checking

It's a technique used to verify that the system fulfills the requirements. It basically represents the possible states a system can be in and the transitions between those states. It's a method to verify complex interactions, concurrent and distributed systems

The requirement of this technique is that we need a model of a system, we have to represent the property in CTL, then we are going to use a model checkup which will tell us whether the specification is true in the particular model. It should satisfy some properties like **Safety Properties,** that is only process can be in a critical section at any time or the **Liveness Properties,** that is when a process wants to enter a critical section it must eventually be allowed to do so.

# Applications of Model Checking

The model checking has a lot of applications in the real world some of which are given below:

### 1 Analysis of Hybrid Control System

It is used for the analysis of Hybrid Control System. Beta-Level tools are used for the design of non-linear control System. For example, discretize Matlab models to create Rulebase inputs or Analysis of a system for tracking and maneuvering targets.

### 2. Avionics

It is also used to ensure the correctness of the avionics system. For example, model checking techniques were used to verify the software used in the Airbus A380 flight control system and also in the Boeing 777.

### 3. Software Engineering

It is also used in Software engineering to verify the correctness of concurrent and distributed systems. For example, it was used to verify the correctness of alternating bit protocol.

## Process of Model Checking

The steps that are followed in the model checking technique are as followed:

**Modelling:**

In this step the system is abstracted and represented as a finite-state automation. It should capture the system in enough details so we can get the specification of its properties

**Specification**

In this step the behavior of the system is expressed in formal language. In this step we basically formalize the properties that must be satisfied by the system.

**Verification**

The third step is verification in which we use the model checker technique on the system and specification to make sure that all the specifications are satisfied by our system. It will either produce a proof that the system satisfies the properties or a counterexample that'll show that the system doesn't satisfy our properties

**Debugging**

In this step we see if the system doesn't satisfy the specification, we identify the source of the problem and try to modify our model and specification accordingly until it satisfies the specifications.

## Model Checking in formal methods

Formal method aims to ensure that the system satisfies its intended behavior and model checking is an essential tool as it allows for automatic verification of systems against their specifications. In formal methods,

model checking is used to verify different properties of a system such as its liveness, safety and correctness properties.

It can also be used to identify the bugs and errors and can guide towards the correcting of these errors and issues. It can also help with large and complex systems to ensure that all their specifications and properties are satisfied

## LTL

It stands for Linear Temporal Logic and is used for the verification of the temporal properties of the system in model checking. It basically describes the properties of a system that changes with time. Some basic modal operators in LTL are:

X: It is “next” - It means the next state in the computation

G: It is “Globally” - Represents a property that’ll hold in all future states

F: “eventually” - It means a property that’ll hold in some future states

U: “Until” - It means a property that holds only until another property becomes true.

It is also used in model checking as it automatically verifies the correctness of reactive systems. It is a widely used temporal logic in the field of computer science.

## CTL

**Computational Tree Logic** focuses on the behavior of systems that evolve over time. It is similar to the LTL, but it is more focused on the

structure of the computation tree.  It can help model check in exploring the system's state space to make sure it satisfies the required property.

## Advantages of Model checking

### Completeness:

It performs a complete analysis of the system under observation. It checks all the behaviors and identifies any errors or bugs in the system. This can be really essential, especially in critical systems where any error can have severe consequences.

### Scalability:

It can handle hardware and software components and can deal with large and complex systems. It can be useful when verifying highly complex systems like embedded and distributed systems.

### Automation:

It is automated that means that it doesn't need any human intervention for any verification. This reduces any risks of human errors and makes sure that large and complex systems can be analyzed more efficiently.

## Disadvantages of Model checking

**State Explosion:**

This problem occurs when as the system grows the number of states grows as well. This can cause the analysis time and memory requirements to get really high.

**Limited Scope:**

Model checking is limited to verifying certain properties like scalability, reliability or performance.

**Property Specification:**

A person will need to properly understand the aspects of the system and also all the specifications it has, and it can be quite hard. Finding out all properties of a system is really hard.

## References


Lynch, N. A., & Tuttle, M. R. (1987). Hierarchical correctness proofs for distributed algorithms. ACM Transactions on Programming Languages and Systems (TOPLAS), 9(3), 356-380.

M. Pezzè et al., “Using model checking to validate flight control software,” in Proc. of the 25th International Conference on Software Engineering, 2003, pp. 872-877.

E. M. Clarke, O. Grumberg, and D. A. Peled, Model Checking, MIT Press, 2000.

J. P. Queille and J. Sifakis, “Specification and verification of concurrent systems in CESAR,” in Proc. of the 5th International Symposium on Programming, 1984, pp. 337-351.

Pnueli, A. (1977). The temporal logic of programs. Proceedings of the 18th Annual Symposium on Foundations of Computer Science (FOCS), 46-57.

Emerson, E. A., & Clarke, E. M. (1981). Using branching time temporal logic to synthesize synchronization skeletons. Science of Computer Programming, 2(3), 241-266.

Khurshid, S., & Sen, K. (2010). "Automated software testing using model checking." Communications of the ACM, vol. 53, no. 9, 2010, pp. 109-117.