# Model Checking, Abstraction–Refinement, and their Implementation

Orna Grumberg
The Technion, Haifa, Israel

Temporal logic model checking is a procedure that accepts a model of a system and a property written in temporal logic. It returns "yes", if the system satisfies the property and "no", otherwise. In the latter case it also provides a counterexample that demonstrates an erroneous behavior of the system.

Model checking procedures typically suffer from high space requirements. Significant amount of research is invested in reducing these requirements. One of the most successful approach is abstraction, which produces a small, abstract model by hiding (abstracting away) some of the system's details. The abstraction is guaranteed to be conservative in the sense that every property true of the abstract model is also true of the concrete, full, model of the system.

Counterexample–Guided Abstraction–Refinement (CEGAR) is a central methodology in application of abstraction. It consists of the following stages:

1. Given a system and a property, build an abstract model of the system.
2. Apply model checking to it.
3. If model checking returns "yes", returns "The system satisfies the property".
4. Otherwise, model checking returns an abstract counterexample. Check if it corresponds to a real counterexample in the system. If it does, return ´"The system does not satisfy the property". If not – refine the abstract model and return to (2).

In recent years, many different implementations of the CEGAR framework have been suggested.

The lectures will first define temporal logics. We will then survey several algorithms for temporal logic model checking. Next, we will present the CEGAR framework and describe different implementations for its different stages.

**References**

1. O. Grumberg. *Abstractions and Reductions in Model Checking*; In: Proof and System Reliability, Proc. of the Marktoberdorf Summer School 2001, H. Schwichtenberg, R. Steinbrüggen (eds), Kluwer, 2002