# 实验九（2）：《Logic in Computer Science — Modelling and Reasoning about Systems》学习报告

## 摘要

本报告概述了 Huth 与 Ryan 《Logic in Computer Science》一书中的主要逻辑体系，重点探讨命题/谓词逻辑、时序逻辑、模态逻辑与 Hoare 逻辑在形式化建模与程序验证中的应用，并结合访问控制与模型检测等案例展示逻辑工具在计算机学科的实践价值。

## 1. 逻辑在计算机科学中的角色

逻辑为计算机科学提供精确定义规范、推理和验证的软件与硬件系统的方法论。形式逻辑语言能够表达不确定行为、并发进程及安全策略，并通过自动化工具（如 SAT/SMT 求解器、模型检查器）支持可计算的验证。

## 2. 基础：命题逻辑与一阶逻辑

命题逻辑使用布尔联结词对基本命题进行组合，适合描述系统状态的真假条件；一阶逻辑引入量词与谓词，可表达对象及其属性之间的关系，为后续逻辑扩展奠定了语法与语义基础。

## 3. 模态与时序逻辑

* 模态逻辑：通过 □（必然）与 ◇（可能）运算符刻画不同可能世界，广泛用于访问控制、安全协议分析。
* 线性时序逻辑 (LTL)：基于时间线对路径进行全局 (G)、最终 (F)、下一时刻 (X) 等运算符的组合，用于规格说明与模型检测。
* CTL 及 CTL\*：在 Kripke 结构上区分路径量词 A/E 与时序运算符，使表达能力覆盖分支未来情况。
* TLA (Temporal Logic of Actions)：适合并发系统与分布式算法的规范。

## 4. Hoare 逻辑与程序验证

Hoare 逻辑以 {P} S {Q} 三元组形式建立程序语句 S 与前/后置条件 P、Q 之间的语义关系，通过推理规则与循环不变式证明程序部分或全部正确性。结合工具化框架（如 Coq, VCG），可自动生成待验证条件以提升可扩展性。

## 5. 其他逻辑体系

* PDL（命题动态逻辑）：将程序视为正则表达式，表达执行路径及其结果属性。
* μ‑演算：利用最小/最大不动点算子统一刻画递归性质，具有很高的表达能力。
* 访问控制逻辑：在模态基础上加入 “says” 与 “speaks‑for” 等运算符，形式化权限委派与证明。

## 6. 应用案例

模型检测工具（如 SPIN、NuSMV）采用 LTL/CTL 形式化规格，对有限状态模型进行自动遍历以发现死锁或安全漏洞；以 Hoare 逻辑为核心的 Verified Software Toolchain 则在 C 语言级别实现了内核模块等安全关键代码的形式化验证；在云计算安全领域，访问控制逻辑被用于证明多租户环境下的授权链条正确性。

## 结论

逻辑框架与自动化证明技术的结合日益成为软件与硬件可靠性保障的基石。掌握多种逻辑及其工具链能够显著提升系统的可验证性与安全性。

## 参考文献

1. [1] Huth M., Ryan M. Logic in Computer Science: Modelling and Reasoning about Systems. Cambridge University Press, 3rd ed., 2022.
2. [2] USTC 课程笔记，《命题逻辑》章节，2021.
3. [3] Zhihu 专栏《计算机科学中的逻辑》系列笔记，2019.
4. [4] Cambridge University Press. Temporal Logics in Computer Science, 2023.
5. [5] Garg D., Abadi M. A Modal Deconstruction of Access Control Logics, FoSSaCS 2008.
6. [6] Cornell University. Hoare Logic, Part I, Course CS4160, 2020.
7. [7] CMU Program Analysis. Lecture 14: Hoare Logic and Verification Condition Generation, 2025.
8. [8] SimpleScience. New Insights in Propositional Dynamic Logic, 2025.
9. [9] SpringerLink. Temporal Logic and State Systems, 2024.