第五章 软件质量保证的自动化方法

5.1.1 什么是形式化方法?

- 形式化方法是基于严格数学基础、对计算机硬件和软件系统进行描述、开发和验证的技术，其数学基础建立在形式语言、语义和推理证明三位一体的形式逻辑系统之上。
- 形式化方法主要研究如何把具有清晰数学基础的模型、规范、分析以及验证融入软硬件开发的各个阶段，是改善和确保计算机系统正确性和可靠性的重要途径。

5.1.2 形式化方法具有哪些优点和不足?

- 优点:
  - **覆盖率高**：形式验证是对指定描述的所有可能的情况进行验证，覆盖率达到了100%
  - **减少开发错误**：采用形式化方法可以极大地减少系统设计开发阶段的错误，避免后期错误修改所带来的高额开销
- 缺点:
  - **实用性不高**：对较小规模项目有效，运用在大规模的软件系统中会出现状态空间爆炸的情况
  - **成本高**：`Mr Nie`没写为啥，大概是较多状态空间与人力物力成本
  - **使用要求高**：形式化方法中的抽象数学符号及理论对软件工程师的使用带来不便，人们要花时间和精力去学习，限制了大多数程序设计人员的学习和使用
  - **不对软件整个生命期负责**：缺乏对软件生命周期内各个阶段提供全面支持的形式化方法，不能确保开发出完全正确的软件，可能会延误项目开发周期、增加开发费用

5.2.1 模型检查方法具有哪些特点和优势?

- 特点:
  - 可以自动执行
  - 在系统不满足性质时提供反例路径
- 优势:
  - 可以在系统开发的各个阶段使用，保障每个阶段的可靠性
  - 在一定成本下可遍历系统状态，可以系统地证明某个目标系统一定不存在某种类型的`bug`

5.2.2 模型检查方法存在哪些问题?

- 仅可以支持有穷系统
- 需要对模型检查所用到的理论有长时间的深入研究
- 实现模型检查器的开发成本高、理论要求高
- 目前对于不同目标系统的通用性较差，故模型检查器的重用性低

5.2.3 模型检查方法有哪些支持工具?

- 符号模型检查工具
- 新符号模型检查工具
- 显示模型检查工具
- SAMC 模型检查工具

5.3.1 什么是定理证明?

- 一种推理技术，其中系统及其特性均用某种逻辑公式表达，这种逻辑由一组公理和推导规则给出

- 一个从系统的公理推导某一特征的证明过程，过程引用了公理和规则，并尽可能采用导出定义和中间引理

5.3.2 定理证明有哪些优点和不足?

- 优点：
  - **高可靠性**：以数理逻辑和类型论的经典研究成果为基础，由计算机自动或半自动地完成并检查证明
  - **可执行性**：能够生成可执行代码获得原型工具，通过运行测例，确认所定义的形式规范的合法性
  - **共享性**：具有类似逻辑基础的不同证明助手之间能够共享证明成果
- 缺点：
  - **并发程序的机械化证明难**：并发程序的验证一直是具有挑战性的难题，而机械化证明的研究成果更为少见
  - **支持并发的面向对象语言的编译器验证难**：虽然出现了一个验证的、支持面向对象语言特性和线程的 *Java* 编译器，但是并没有考虑字节码到本地二进制码的正确翻译
  - **机械化定理证明的开发成本较高**：使用和掌握比较困难

5.4.1 仿真的作用是什么?

- 给出了一个成本低、效率高的方法来消除软硬件代价昂贵并且存在潜在危险的设计缺陷
- 使设计者深入了解难以测量或无法测量的系统特性

5.5.1 容错计算的意义是什么?

- 保证了软件的失效率，保证了良好的鲁棒性和良好的用户体验，从而保证了设计出来的软件的质量
- 对开发者，容错计算技术的加入提升了前期软件项目编写的全面性，且能使得后期的错误维护等工作更加方便，提升开发效率；避免了生产生活中出现的极其严重的失误导致的损失，维护了开发者的利益
- 对使用者，容错计算技术的加入增加了应用程序的鲁棒性，给用户的反馈更加直接且快捷，提升了使用软件的舒适性。
- 对社会，维护了社会的稳定发展，，越来越可靠的系统的出现会使人类接受更智能化的生活，稳定的社会需求需要我们使用容错计算来提高软件的保障性能

5.5.2 容错计算有哪些主要的方法?

- 软件容错方法：
  - 多处理器和特别设计的操作系统来达到容错
  - 防卫式程序设计
  - 恢复块方法
  - *N* 版本程序设计
- 硬件容错方法：
  - *Stratus*
  - 磁盘镜像
  - 数据重读
- 如果你觉得这道题狗屁不通，这不是我的问题，我怀疑*Mr Nie*并不知道自己在写什么