《编译技术》

第九、十章：中间代码优化与目标代码生成

学　号 22920212204396

姓　名 黄子安

2024 年 6 月 1 日

1、何谓代码优化？进行优化所需要的基础是什么？

优化：编译时为改进目标程序的质量而进行的各项工作，包括提高时间效率和空间效率

基础：必须是**等价变换**且为**优化的努力必须是值得的**，此外要了解目标硬件平台的架构和性能特点，以便进行针对性的优化

2、编译过程中可进行的优化如何分类？

按作用范围分类：

- 局部优化：在基本块内进行优化，如常量传播、强度削弱、公共子表达式消除等。
- 全局优化：跨基本块在函数或方法范围内进行优化，如循环不变代码外提、全局公共子表达式消除、寄存器分配等。
- 过程间优化：跨函数或方法进行优化，如内联扩展、跨过程的常量传播、跨过程的寄存器分配等。

按执行阶段分类：

- 前端优化：在编译的早期阶段进行的优化，如语法树转换、类型检查、初步的常量折叠等。
- 中端优化：在中间表示（IR）阶段进行的优化，如数据流分析、循环优化、指令合并等。
- 后端优化：在生成目标代码时进行的优化，如指令调度、寄存器分配、目标代码优化等。

按优化目标分类：

- 速度优化：提高程序的执行速度，如循环展开、函数内联等。
- 空间优化：减少程序的内存占用，如删除死代码、数据压缩等。
- 能耗优化：降低程序的能耗，通常与速度优化和空间优化有关。

3、最常用的代码优化技术有哪些？

- 常量传播：将程序中所有能确定的常量值在编译时传播到使用这些常量的地方。
- 常量折叠：在编译时计算出常量表达式的值，以减少运行时的计算量。
- 公共子表达式消除：识别并消除程序中重复出现的子表达式，以减少计算次数。
- 无用代码消除：移除程序中不会被执行的代码，减少不必要的资源消耗。
- 循环优化：包括循环展开、循环合并、循环不变代码外提等，目的是提高循环的执行效率。
  - 循环展开：减少循环控制开销，增加指令级并行度。
  - 循环不变代码外提：将循环中不变的计算移到循环外，提高效率。
- 内联扩展：将函数调用替换为函数体，从而减少函数调用的开销。
- 寄存器分配：有效利用 CPU 寄存器，减少内存访问次数。
- 指令调度：重新排列指令顺序，以减少处理器流水线的停顿，增加指令并行度。
- 强度削弱：将高代价的运算替换为低代价的运算，如将乘法替换为加法。
- 代码移动：将频繁执行的代码移到执行频率较低的地方，以减少重复计算。