Post Link-Time Optimization on the Intel IA-32 Architecture*

Benjamin William Schwarz

Abstract

Post link-time optimization of executables has been investigated by several projects in recent years. These optimization systems have targeted RISC architectures like the Compaq Alpha, and have shown that there is considerable room for improvement in compiler-generated code. Classical compiler optimizations like constant propagation, function inlining, and dead code elimination have been shown to be relatively effective when applied at link-time. In addition, other optimizations—such as value specialization, load/store forwarding, and code layout—that are not typically carried out at compile-time can also be used effectively. Unfortunately, many of the analyses introduced by other systems are insufficient when carried out on a CISC machine (e.g. the x86). We describe PLTO, a link-time optimizer for the Intel IA-32 architecture, that addresses the inherent difficulties in static analysis of binaries compiled for a CISC architecture. Many of the challenging issues stem from intrinsic characteristics of the architecture, such as the small register set which lends way to a heavy reliance on using the runtime stack. This paper discusses many analyses and optimizations used by PLTO, and we show the performance gains our system is able to achieve over compiler-generated, heavily-optimized executables.

*This research was supported by the National Science Foundation through grants ACR-9720738 and CCR-0113633.