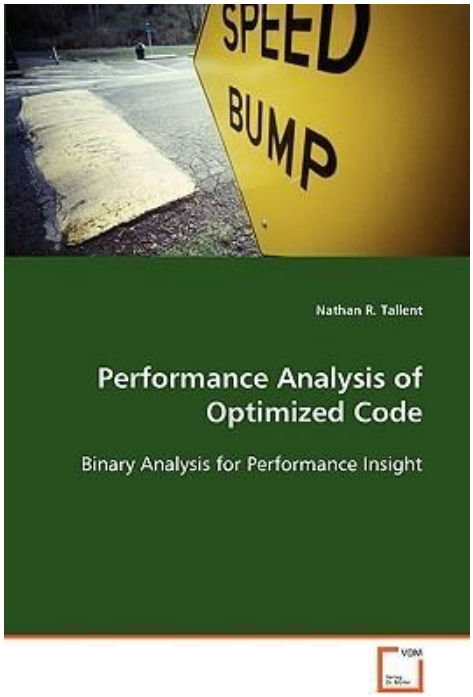

Performance Analysis of Optimized Code

Nathan R Tallent

Modern applications frequently employ sophisticated object-oriented design. In these codes, deep loop nests are often spread across multiple routines. To achieve high performance, such codes rely on compilers to inline routines and optimize loops. Consequently, to effectively interpret performance, transformed loops must be understood in the calling context of transformed routines. *** To understand the performance of optimized object-oriented code, we describe how to analyze optimized object code and its debugging sections to recover its program structure and reconstruct a mapping back to its source code. Using this mapping, we combine the recovered static program structure with dynamic call path profiles to expose inlined frames and loop nests. Experiments show that performance visualizations based on this information provide unique insight into the performance of complex object-oriented codes written in C++. *** This work should be of interest to performance tools developers and, more broadly, application developers who care about performance. It is implemented in Rice Universitys HPCToolkit, a performance analysis toolkit.