This talk describes the Broadway compiler’s approach to analyzing and optimizing software libraries. We argue that the right approach is to integrate domain-specific information with traditional optimization facilities. We outline three classes of optimizations that can be performed with such a system, and we briefly present performance results to show that such a system can produce significant performance improvements.

We first argue that in order to optimize software libraries, superficial syntactic information is insufficient. Instead, we need a solution that mimics the deep analyses that are required of traditional compiler optimizations. In particular, program analysis is needed to perform data-flow analysis, create control flow graphs, and perform pointer and dependence analysis. We then explain how the optimization of software libraries presents unique challenges, since libraries are essentially lightweight domain-specific languages that are embedded in some base programming language. Thus, any solution needs to understand both languages and the interactions between them. After briefly explaining how our annotation-based system meets these challenges, we conclude by presenting some performance results and directions for future work.