OPTIMIZATION SOFTWARE CLASS LIBRARIES

Edited by
Stefan Voß and David L. Woodruff

Optimization problems in practice are diverse and evolve over time, giving rise to requirements both for ready-to-use optimization software packages and for optimization software libraries, which provide more or less adaptable building blocks for application-specific software systems. In order to apply optimization methods to a new type of problem, corresponding models and algorithms have to be “coded” so that they are accessible to a computer. One way to achieve this step is the use of a modeling language. Such modeling systems provide an excellent interface between models and solvers, but only for a limited range of model types (in some cases, for example, linear) due, in part, to limitations imposed by the solvers. Furthermore, while modeling systems especially for heuristic search are an active research topic, it is still an open question as to whether such an approach may be generally successful. Modeling languages treat the solvers as a “black box” with numerous controls. Due to variations, for example, with respect to the pursued objective or specific problem properties, addressing real-world problems often requires special purpose methods. Thus, we are faced with the difficulty of efficiently adapting and applying appropriate methods to these problems. Optimization software libraries are intended to make it relatively easy and cost effective to incorporate advanced planning methods in application-specific software systems.

A general classification provides a distinction between callable packages, numerical libraries, and component libraries. Component libraries provide useful abstractions for manipulating algorithm and problem concepts. Object-oriented software technology is generally used to build and apply corresponding components. To enable adaptation, these components are often provided at source code level. Class libraries support the development of application-specific software systems by providing a collection of adaptable classes intended to be reused. However, the reuse of algorithms may be regarded as “still a challenge to object-oriented programming”. Component libraries are the subject of this edited volume. Within a careful collection of chapters written by experts in their fields discusses all relevant aspects of component libraries. To allow for wider applicability, we restrict the exposition to general approaches as opposed to problem-specific software.