ABSTRACT

Design patterns capture the essentials of recurring best practice in an abstract form. Their merits are well established in domains as diverse as architecture and software development. They offer significant benefits, not least a common conceptual vocabulary for designers, enabling greater communication of high-level concerns and increased software reuse. Inspired by the success of software design patterns, this chapter seeks to promote the merits of a pattern-based method to the development of metaheuristic search software components. To achieve this, a catalog of patterns is presented, organized into the families of structural, behavioral, methodological and component-based patterns. As an alternative to the increasing specialization associated with individual metaheuristic search components, the authors encourage computer scientists to embrace the ‘cross cutting’ benefits of a pattern-based perspective to optimization algorithms. Some ways in which the patterns might form the basis of further larger-scale metaheuristic component design automation are also discussed.
BACKGROUND

In recent years, modern optimization algorithms have attracted a growing number of scientists, decision makers and practitioners. Indeed, powerful intelligent computational techniques such as metaheuristics have emerged for solving a vast number of complex real-world problems, exploiting both optimization theory and practice. Increasingly, metaheuristic optimization techniques offer generic, flexible, robust, and versatile frameworks for solving complex problems of optimization and search in real-world application areas such as economics and engineering. Many metaheuristics - evolutionary algorithms, particle swarms, ant colonies, to name a few - are population-based, which makes them particularly robust and applicable to a diverse range of application domains. Nevertheless, as is the case with many other algorithms, tailoring the design of larger-scale metaheuristic search techniques, components and frameworks can be complex and decidedly non-trivial and may raise cross-cutting concerns that are critical for system performance.

Previously, concerned with a diversity of buildings’ architecture, Alexander (1979) advocated a ‘timeless way of building’, drawn from thousands of years of traditional construction. Central to this approach is a ‘quality without a name’: good architectural quality is something that can be recognized, but difficult to describe in words, i.e. ‘you know it when you see it’. Alexander suggested that this quality of a design could be captured in terms of patterns such that “each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution in such a way that you can use this solution a million times over, without ever doing it the same way twice”. (Alexander et al., 1977, p. x).

Inspired by this notion of ‘heuristics at a broad, architectural scale’, Gamma, Helm, Johnson, & Vlissides (1995) applied design patterns to software design, an approach which revolutionized software development. Defined as “… descriptions of communicating objects and classes that are customized to solve a general design problem in a particular context”, they proposed that design patterns in software should comprise four essential elements: a pattern name, a design problem, a design solution and the consequences, i.e. the results and tradeoffs of applying the pattern. According to Gamma et al., “These patterns solve specific design problems and make object-oriented designs more flexible, elegant, and ultimately reusable” (Gamma et al., 1995, p. 1). Design patterns prompted widespread change in the accepted practice of software design, leading to component descriptions at a more abstract architectural level. Subsequently, pattern-oriented software architectures have been proposed, which “…present, discuss and contrast and relate the many known flavors and applications of the pattern concept: stand-alone patterns, pattern complements, pattern compounds, pattern stories, pattern sequences, and … pattern languages”. (Buschmann, Henney, & Schmidt, 2007, p. xxxii).

Applying design patterns offers many benefits. For example, reuse of successful designs and architectures is easier, since “expressing proven techniques as design patterns makes them more accessible to developers of new systems” (Gamma et al., 1995, p. 2). To address the complexity and associated non-trivial issues of designing and applying metaheuristics, this chapter advocates a pattern-based perspective on optimization architectures, with a particular focus on the construction of larger scale frameworks.

Firstly, the motivation for metaheuristic search design patterns is discussed and the format used is described (Motivation section). A catalog of a dozen patterns is then presented, organized into structural, behavioral, methodological and component-based categories (in their own sections respectively). Looking towards possible future directions for metaheuristic design patterns (in the section Future Direction), we
Related Content

Utilizing the Modified Self-Adaptive Differential Evolution Algorithm in Dynamic Cellular Manufacturing System
[www.igi-global.com/article/utilizing-modified-self-adaptive-differential/67330?camid=4v1a](www.igi-global.com/article/utilizing-modified-self-adaptive-differential/67330?camid=4v1a)

A Hybrid Algorithm Based on Monte-Carlo Simulation for the Vehicle Routing Problem with Route Length Restrictions
[www.igi-global.com/chapter/hybrid-algorithm-based-monte-carlo/58519?camid=4v1a](www.igi-global.com/chapter/hybrid-algorithm-based-monte-carlo/58519?camid=4v1a)

Supply Chain Network Design Using an Enhanced Hybrid Swarm-Based Optimization Algorithm
[www.igi-global.com/chapter/supply-chain-network-design-using-an-enhanced-hybrid-swarm-based-optimization-algorithm/147511?camid=4v1a](www.igi-global.com/chapter/supply-chain-network-design-using-an-enhanced-hybrid-swarm-based-optimization-algorithm/147511?camid=4v1a)

An Integrated Model of Data Envelopment Analysis and Artificial Neural Networks for Improving Efficiency in the Municipal Solid Waste Management