Variability is a fundamental property of software systems that can be customized for different market segments or contexts of use by reusing predefined, adjustable artifacts (e.g., requirements, architecture, components, code, and tests). Variability is ubiquitous in the design, development, and deployment of software systems in many domains, such as aviation, automobiles, integrated circuits, and mobile phones. For example, the software deployed on a mobile phone may optionally support video calls. However, variability introduces essential complexity of managing a large number of variants and diverse artifacts involved. A major challenge is to accurately and efficiently find the optimal variant that meets the functional and quality (e.g., performance, cost, and energy consumption) requirements of stakeholders. To address this challenge, I aim at variability-aware system engineering, involving variability modeling, quality prediction, and combinatorial optimization. The overarching objective is to investigate methods, techniques, and tools for developing adaptable and reliable software systems to meet various and frequently changing stakeholder requirements [1].

Variability modeling abstracts system functionalities relevant to stakeholders as features, and it manages features, constraints, and related artifacts using systematic methodologies from model-driven engineering and product-line engineering [6]. It enables system mass customization to reduce development cost, enhance quality, and shorten time to market. Variability models can be conveniently specified in propositional logic, which enables automated reasoning (e.g., validating a system variant) using off-the-shelf satisfiability solvers. However, variability models and related artifacts are subject to change. To this end, I proposed an approach based on change propagation and ontology evolution to support the consistent evolution of large-scale variability models [5]. Furthermore, I investigated the real evolution history of Linux Kernel and explored the evolution rules [4]. Further research plans include: (1) investigating the evolution histories of other systems, such as eCos, Android, and FreeBSD, and (2) identifying the patterns of variability evolution. The long-term research objective is to discover effective mechanisms that maintain sustainable evolution of large-scale, variant-rich systems, which complements existing methodologies of software evolution and maintenance.

Quality prediction mines the correlations between features and quality attributes of systems, and determines the impact of features and feature interactions on quality attributes. Implementing and measuring all system variants is a straightforward way to acquire accurate quality attributes, but it is often infeasible, because there is often an exponential number of variants, and the effort of implementing and measuring a variant may be high. To this end, I proposed a quality-prediction approach via statistical machine learning that uses a small random sample of measured variants as a basis to accurately predict the quality attributes of other variants [3]. Empirical results on six real-world case studies demonstrate an average prediction accuracy of 94% based on only small random samples. Moreover, the proposed approach works progressively, such that one can use it to produce predictions, starting with a small random sample, and subsequently improve the prediction accuracy when further measurements are available. Further research plans
include: (1) applying the proposed approach to automotive and avionics systems, such as predicting the response time and reliability of Controller Area Network (CAN) messages, (2) investigating effective sampling schemes and efficient parameter-tuning techniques to guarantee high prediction accuracy, and (3) studying adaptive experimental design for robust performance prediction across different hardware environments. The long-term research objective is to combine active learning, systematic parameter tuning, and adaptive experimental design to find a “representative” sample of measured variants and efficiently reach the sweet spot between prediction accuracy and measurement effort.

Combinatorial optimization explores a finite search space of variants, and finds the one that is optimal regarding single objective or the ones that are Pareto-optimal regarding multiple (often conflicting) objectives. For example, a system architect may request a system design with low cost and high performance. Combinatorial optimization is a fundamental challenge in many problems in software engineering (e.g., architecture design, test data generation, and project planning) and other domains (e.g., hybrid vehicle powertrain design, electric vehicle battery design, and civil infrastructure repair planning). Most combinatorial optimization problems are NP-hard. To solve them efficiently, approximate approaches that depend mainly on meta-heuristics, such as the genetic algorithm I proposed in the past [7], have been advocated for years. They solve optimization problems in an acceptable time, but they find only near-optimal solutions, and often suffer from parameter sensitivity (i.e., the accuracy of the found solutions varies widely with the parameter settings of these approaches). In contrast, exact methods that scan all candidate solutions one by one often take too long for large-scale problems, but they are accurate in finding all, exact optimal solutions, which is desirable for those stakeholders who never want to miss any optimal opportunity. To this end, I proposed a number of exact, parallel algorithms that solve combinatorial optimization problems accurately and efficiently [2]. Empirical results on three case studies demonstrate that the proposed parallel algorithms achieve substantial (even super-linear) speedups that scale well up to 64 cores. Further research plans include: (1) applying the proposed algorithms to automotive systems, such as optimizing vehicle wire harness design; (2) scaling exact, parallel optimization algorithms to larger problems and more cores; and (3) combining exact, parallel optimization algorithms with existing approximate optimization methods. The long-term research objective is to devise accurate and fast combinatorial-optimization algorithms that can guarantee the (approximate) performance bounds, and to apply them to various industrial problems.