
Vincent A. Cicirello

1 Computer Science, School of Business, Stockton University, Galloway, NJ 08205

Summary

Discrete optimization problems are often of practical real-world importance as well as computationally intractable. For example, the traveling salesperson, bin packing, and longest common subsequence problems are NP-Hard, as is resource constrained scheduling, and many single-machine scheduling problems (Garey & Johnson, 1979). Polynomial time algorithms for such problems are unlikely to exist, and the best known algorithms that guarantee optimal solutions have a worst-case exponential runtime. It is thus common to use stochastic local search and other metaheuristics (Gonzalez, 2018). Stochastic local search algorithms begin at a random search state, and apply a sequence of neighbor transitions to nearby search states. This includes perturbative (Hoos & Stützle, 2018) algorithms like simulated annealing (DelaHaye, Chaimatanan, & Mongeau, 2019) and hill climbers (Hoos & Stützle, 2018), where each search state is a complete candidate feasible solution, and a mutation operator makes a small random modification to move to another local candidate solution; and also includes constructive (Hoos & Stützle, 2018) algorithms like stochastic samplers (Bresina, 1996; Cicirello & Smith, 2005; Grasas, Juan, Faulin, de Armas, & Ramalhinho, 2017; Langley, 1992; Reyes-Rubiano, Calvet, Juan, Faulin, & Bové, 2020), where each search state is a partial solution that is iteratively transformed into a complete solution. Stochastic local search algorithms do not guarantee optimal solutions. However, they often find near-optimal solutions in much less time than systematic search. They also offer an anytime property (Jesus, Liefooghe, Derbel, & Paquete, 2020; Zilberstein, 1996), where solution quality improves with runtime.

Chips-n-Salsa is a Java library of customizable, hybridizable, iterative, parallel, stochastic, and self-adaptive local search algorithms. Its focus is discrete optimization, but also supports continuous optimization. The library provides a variety of solution representations, including BitVector and IntegerVector classes, indexable vectors of bits and integers, respectively, along with corresponding mutation operators. The library utilizes our significant prior research on permutation optimization problems, providing an extensive set of mutation operators for permutations (Cicirello, 2016; Cicirello & Cernera, 2013), including window-limited mutation (Cicirello, 2014). It uses the JavaPermutationTools (JPT) (Cicirello, 2018a) library for efficiently representing solutions to such problems. For continuous problems, Chips-n-Salsa provides a RealVector class, Gaussian mutation (Petrowski & Ben-Hamida, 2017), Cauchy mutation (Petrowski & Ben-Hamida, 2017), and uniform mutation. The library includes optimization problems useful for benchmarking metaheuristic implementations, such as the well-known OneMax problem (Doerr & Neumann, 2019), BoundMax (generalization of OneMax to integers), the Permutation in a Haystack problem (Cicirello, 2016), and polynomial root finding.

The repository (https://github.com/cicirello/Chips-n-Salsa) contains the library source code, and programs with examples of key functionality. API and other documentation is hosted on the web (https://chips-n-salsa.cicirello.org/). The library can be integrated into projects using maven via GitHub Packages.
Chips-n-Salsa Features and Functionality:

- **Stochastic Local Search Algorithms**: The library supports simulated annealing (De-lahey et al., 2019), with all of the common annealing schedules, as well as advanced annealing schedules. It includes common forms of hill climbing (Hoos & Stützle, 2018), and several stochastic sampling (Grasas et al., 2017) algorithms, such as iterative sampling (Langley, 1992), Heuristic Biased Stochastic Sampling (HBSS) (Bresina, 1996), Value Biased Stochastic Sampling (VBSS) (Cicirello, 2003; Cicirello & Smith, 2005), and stochastic sampling with acceptance bands (Gomes, Selman, & Kautz, 1998; Oddi & Smith, 1997). We optimized random number generation based on our prior research to minimize the runtime impact of one of the more costly operations of a metaheuristic (Cicirello, 2018b).

- **Customizable**: Chips-n-Salsa uses generic types enabling using the library to optimize other representations beyond what is provided in the library. It also enables easily integrating new mutation operators, and customizing all stages of the local search (e.g., choice of annealing schedule for simulated annealing).

- **Hybridizable**: Chips-n-Salsa supports integrating multiple forms of local search (e.g., hybrids of hill climbing with simulated annealing), creating hybrid mutation operators (e.g., combining multiple mutation operators), and running more than one type of search for the same problem concurrently using multiple threads as a form of algorithm portfolio (Gomes & Selman, 2001; Tong, Liu, & Yao, 2019).

- **Iterative**: Chips-n-Salsa supports multistart metaheuristics, including several restart schedules (Cicirello, 2017; Luby, Sinclair, & Zuckerman, 1993) for varying the run lengths across the restarts.

- **Parallel**: Chips-n-Salsa enables parallel execution of multiple instances of the same, or different, stochastic local search algorithms for a problem instance to accelerate search by exploiting multicore architectures (Cicirello, 2017).

- **Self-Adaptive**: Chips-n-Salsa includes adaptive annealing schedules (Hubin, 2019; Štefankovič, Vempala, & Vigoda, 2009) for simulated annealing, such as Modified Lam (Boyan, 1998; Cicirello, 2007), self-tuning variations of the simpler annealing schedules, and adaptive restart schedules (Cicirello, 2017).

Statement of Need

The target audience of Chips-n-Salsa includes those conducting computational research in diverse domains, wherever applications of NP-Hard optimization problems is important. Existing local search open source implementations are often intimately tied to a specific problem, and usually a specific form of local search. For example, a GitHub search finds countless solvers for problems like Boolean Satisfiability or the Traveling Salesperson. Such problem-dependent implementations are not easily adapted to new problems. There are notable exceptions. In particular, ECJ (Scott & Luke, 2019) and Jenetics (Jenetics, 2020) are both mature and well-maintained Java libraries for genetic algorithms and related forms of evolutionary computation. They are problem-independent, support multiple representations, and include multithreaded capabilities. Their focus, however, is on population-based evolutionary search; whereas Chips-n-Salsa is focused on single-solution algorithms. Another especially noteworthy project is emili (Pagnozzi & Stützle, 2019), which is a C++ framework supporting hybrid, single-solution, stochastic local search algorithms. Chips-n-Salsa is problem-independent, representation-independent, and supports hybrid local search, as well as parallel execution to easily exploit multicore architectures to speed up problem solving. Furthermore, the self-adaptive and self-tuning features streamline development, eliminating the need for the developer to tune control parameters.
References


