Portfolios of Genetic Algorithms

Alex S. Fukunaga
Computer Science Department
University of California, Los Angeles
fukunaga@cs.ucla.edu

Abstract

We propose an *anytime algorithm portfolio* technique which allocates computational resources among sets of control parameter value settings for evolutionary algorithms. Meta-level optimization of the portfolios is enabled by applying a bootstrap sampling approach to a database of individual algorithm performance on instances from a problem distribution. Experiments with a genetic algorithm portfolio applied to the traveling salesperson (TSP) domain show that the portfolio approach can yield better performance on a distribution of problem instances than the best single control parameter set.

Let A be an optimization algorithm, d a problem instance from problem class D, and T a resource usage limit for A (measured in number of objective function evaluations). Let $U(A, d, T)$ be the best-so-far utility of the algorithm A on d after T steps. An *anytime algorithm portfolio* (an extension of the framework proposed in [3])is a set $P = \{(A_1, t_1), ...(A_n, t_n)\}$, where $\sum_{i=1}^{n} t_i = T$, and $A_1, ...A_n$ are anytime algorithms that can be applied to the problem instance. For example, $A_1, ..A_n$ can be sets of control parameter values for an evolutionary algorithm, where, e.g., A_1 could be the parameter set $(population = 100, mutateRate = 0.01)$, and A_2 is the parameter set $(population = 200, mutateRate = 0.05)$. P is executed as follows: An independent run of A_i is executed on d for t_i steps for each i, $1 \leq i \leq n$; the best solution found among all of the runs is returned. The portfolio utility $U(P, d, T)$ is $max(U(A_1, d, t_1), ...U(A_n, d, t_n))$. We seek a portfolio which maximizes expected utility $E[U(P, D, T)]$.

When A is executed on instance $d \in D$ for mq (m and q are integers)iterations, the score of the best-so-far solution at every q iterations is saved in a *performance database* (PDB) entry, $DB(A, d, runID) = \{(q, u_1), (2q, u_2), ...(mq, u_m)\}$, The PDB can serve as an empirical approximation of the random variables $\mathcal{U}_{Ad} = \{U(A, d, q), ...U(A, d, mq)\}$. Data from runs on different problem instances and the distribution of instances can be combined to yield approximations for $U(A, D, t)$. A heuristic search algorithm is used to find a portfolio with maximal expected utility for D, where the utility metric is a weighted sum of the mean and variance of the portfolio performance. Candidate portfolios are evaluated by estimating their expected utility via sampling (with replacement) from the PDB, i.e., we generate a bootstrap sampling distribution [1] and compute its mean and variance. This bootstrap-based portfolio evaluation process is orders of magnitude faster than actually executing the portfolio, enabling fast meta-level optimization of the portfolio. A similar approach was used in [2] to optimize static restart strategies (a special case of anytime algorithm portfolios).

A PDB was generated for 54 genetic algorithm control parameter sets applied to 10 training problems from a distribution D_{tsp} of symmetric TSP instances. An optimal portfolio for distribution was generated using the method described above, and its performance (mean and variance on 30 independent runs) was compared to the performance of the best single control parameter set for each of 10 new test problems from D_{tsp}. The portfolio significantly outperformed the best single control parameter set on 7 of the 10 test problems; performance was not statistically different on 1 problem, and the portfolio was outperformed by the best single control parameter set on 2 problems.

References

[1] B. Efron and R. Tibshirani. Bootstrap methods for statistical errors, confidence intervals, and other measures of statistical accuracy. *Statistical Science*, 1(1):54–47, 1986.

[2] A. Fukunaga. Restart scheduling for genetic algorithms. In *Proc. Parallel Processing from Nature (PPSN)*, pages 357–358, 1998.

[3] B.A. Huberman, R.M. Lukose, and T. Hogg. An economics approach to hard computational problems. *Science*, 275(5269):51–4, January 1997.