Restart strategy for biased random-key genetic algorithms

Mauricio G. C. Resende[1], Luis Morán-Mirabal[2], José Luis González-Velarde[2], Rodrigo F. Toso[3]

[1] AT&T Labs Research
Florham Park, NJ, USA
mgcr@research.att.com

[2] Tecnológico de Monterrey
Monterrey, Mexico
{lmoran,gonzalez.velarde}@itesm.mx

[3] Rutgers University
Piscataway, NJ, USA
rtoso@cs.rutgers.edu

Abstract

A biased random-key genetic algorithm (BRKGA) encodes solutions to an optimization problem as a vector of random-keys, i.e. a vector of real numbers in the interval $[0, 1)$, each generated independently and uniformly at random. A BRKGA evolves a population of random-keys applying a simple evolutionary strategy in which pairs of vectors from one generation are combined to produce offspring solutions of the next generation. An offspring is more likely to inherit the keys of its most fit parent. Strategies to avoid getting trapped in locally optimal solutions include increasing randomness in the algorithm and population restart. A BRKGA can be imbedded in a multi-start strategy in which a population is evolved for a number of iterations and then is thrown away so the evolutionary process can be restarted with a fresh population. Such restart points are triggered by a restart strategy. We describe a restart strategy for biased random-key genetic algorithms. A computational experiments illustrates the speed-up that can be achieved by applying such a strategy.

1 Biased random-key genetic algorithms

Biased random-key genetic algorithms (BRKGA) [3] are based on the random-key genetic algorithm (RKGA) of Bean [2]. Both evolutionary algorithms encode solutions as vectors of n random keys and evolve a population of p vectors of random keys. A *random key* is simply a real number generated uniformly at random in the interval $[0, 1)$. A *decoder* is a deterministic algorithm that takes as input a vector of n random keys and returns a feasible solution of the optimization problem and its cost or fitness.

At each iteration k, the random-key vectors are decoded and the population is partitioned into a smaller elite set of p_e vectors made up of the most fit solutions and a larger set of $p - p_e$ non-elite solutions. Population $k+1$ contains all p_e elite-set vectors of population k as well as p_m newly-generated mutants. A *mutant* is simply a vector of n-random keys and its role is to help the algorithm avoid getting trapped in locally optimal solutions. The remaining $p - p_e - p_m$ solutions are produced by combining pairs of solutions (vectors a and b) from population k using uniform parameterized crossover [7], where the i-th key $c(i)$ of the offspring inherits the the i-th key of parent a, i.e. $c(i) = a(i)$ with probability ρ_a and the i-th key of parent b with probability $\rho_b = 1 - \rho_a$. RKGAs and BRKGAs differ in how parents are selected for mating and how crossover is applied. In a RKGA both parents are selected at random from the entire population, the first denoted by a and the second by b. In a BRKGA parent a is always selected from the elite set and parent b from the non-elite set. Furthermore, in a BRKGA $\rho_a > 0.5$, i.e. the child has a greater probability of inheriting the key of its elite parent than that of its non-elite parent. As a result, a BRKGA is almost always more efficient than a RKGA [4].

2 Restart strategy

An effective global optimization algorithm must have a mechanism to hinder entrapment in locally optimal solutions. In a BRKGA, the mutants play this role. By changing the value of parameter p_m one

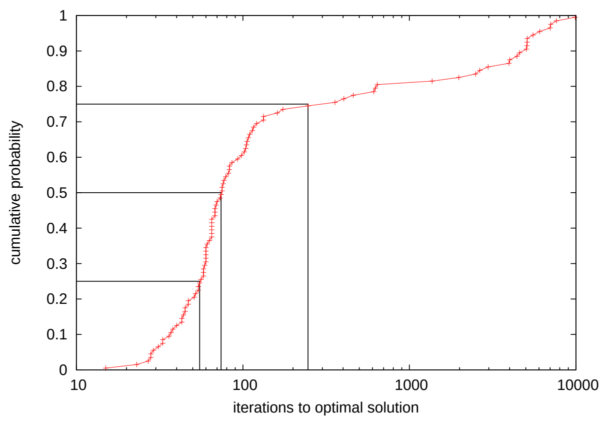

Figure 1: Iteration count distribution of a BRKGA to find an optimal covering of Steiner triple covering problem `stn243` [6].

controls the amount of noise in the population. The larger the value of p_m, the greater the amount of randomness in the population and the less likely it becomes that the population will converge to one or more locally optimal solutions. Another solution is to adopt a *restart strategy*. We propose a restart strategy for BRKGA that is similar to the one described by Resende and Ribeiro [5] for GRASP with path-relinking. This strategy, called $restart(k)$ restarts the BRKGA every k iterations without improvement in the value of the best found solution. It does this by emptying out the population and reloading it with newly-generated random-key vectors. The incumbent over all restarts is saved for output at the conclusion of the run.

Consider the iteration count distribution of a BRKGA of [6] to find an optimal solution of Steiner triple covering problem `stn243` in Figure 1. The iteration-to-target plot [1] is generated by running a BRKGA 100 times, each using a different seed for the random number generator and recording the number of iterations taken by the algorithm to find an optimal solution. The figure shows that 25% of runs took no more than 55 iterations, 50% took no more than 74, and 75% took no more than 246. However, 10% took more than 4597 iterations, 5% took more than 5532 iterations, 2% more than 7061, and the longest run took 9903. Let I be the random variable number of iterations to target solution. For `stn243`, Figure 1 suggests that $\Pr(I \geq 246) \approx 1/4$. By restarting after 246 iterations and assuming independence runs, $\Pr(I \geq 492 \mid I \geq 246) \approx 1/4$. Therefore, $\Pr(I \geq 492) = \Pr(I \geq 246) \times \Pr(I \geq 492 \mid I \geq 246) \approx 1/4^2$. It is easy to show by induction that the probability that the algorithm will still be running after k cycles of 246 iterations is approximately $1/4^k$. For example, the probability the algorithm with restart will still be running after 1230 iterations (five cycles of 246 iterations between restarts) is approximately $1/4^5 = 1/1024 \approx 0.1\%$. This is in contrast to the estimated 80% probability the algorithm without restart will run for over 641 iterations. This is an example of a slightly different restart strategy than the one we propose. In this strategy restarts are done every k iterations while in the more robust strategy proposed in this paper restarts are done every k iterations without improvement of the incumbent. We discuss this issue of robustness in greater detail in the full paper.

3 Numerical example

As an illustration of our strategy consider the BRKGA of Resende et al. [6] for the Steiner triple covering problem, a type of set covering problem. In set covering, we are given n finite sets $P_1, P_2, \ldots, P_n$. Let sets I and J be defined as $I = \cup_{j=1}^{n} P_j = \{1, 2, \ldots, m\}$ and $J = \{1, \ldots, n\}$. A subset $J^* \subseteq J$ is called a *cover* if $\cup_{j \in J^*} P_j = I$. The *set covering problem* is to find a cover of minimum cardinality. The characteristics of sets $P_1, P_2, \ldots, P_n$ make the Steiner triple covering problem a computationally difficult set covering problem. See [6] for a review of the literature of Steiner triple covering.

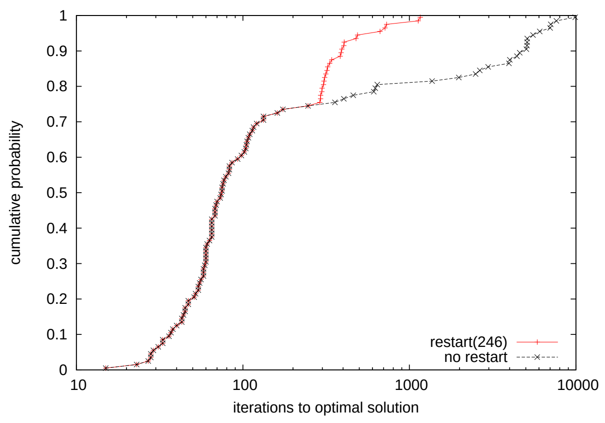

Figure 2: Iteration count distributions of BRKGAs with and without restart on Steiner covering problem `stn243`.

Figure 2 illustrates how the restart strategy can improve the performance of a BRKGA. The figure shows in red the iteration count distribution to a best known solution of `stn243` of a BRKGA with a $\mathit{restart}(246)$ strategy and in black the same BRKGA, but with no restart. While the BRKGA with restart exhibits behavior similar to what we estimated in the previous section and never took over 1162 iterations to find a solution, in 19 of 100 runs the BRKGA without restart took over 1370 iterations. With respect to the variant without restart, $\mathit{restart}(246)$ reduced the average number of iterations from 1003.89 to a mere 166.81 and the standard deviation from 206.80 to only 20.75.

4 Conclusion

In the full paper, we will discuss the choice of parameter k in $\mathit{restart}(k)$ to produce a robust heuristic and report on extensive computational experiments with several implementations of biased random-key genetic algorithms for different optimization problems.

References

[1] R.M. Aiex, M.G.C. Resende, and C.C. Ribeiro. TTTPLOTS: A perl program to create time-to-target plots. *Optimization Letters*, 1:355–366, 2007.

[2] J. C. Bean. Genetic algorithms and random keys for sequencing and optimization. *ORSA J. on Computing*, 6:154–160, 1994.

[3] J.F. Gonçalves and M.G.C. Resende. Biased random-key genetic algorithms for combinatorial optimization. *J. of Heuristics*, 17:487–525, 2011.

[4] J.F. Gonçalves, M.G.C. Resende, and R.F. Toso. Biased and unbiased random-key genetic algorithms: An experimental analysis. Technical report, AT&T Labs Research, Florham Park, New Jersey, 2012.

[5] M.G.C. Resende and C.C. Ribeiro. Restart strategies for GRASP with path-relinking heuristics. *Optimization Letters*, 5:467–478, 2011.

[6] M.G.C Resende, R.F. Toso, J.F. Gonçalves, and R.M.A Silva. A biased random-key genetic algorithm for the Steiner triple covering problem. *Optimization Letters*, 6:605–619, 2012.

[7] W. M. Spears and K. A. DeJong. On the virtues of parameterized uniform crossover. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 230–236, 1991.