

# LKH's Genetic Algorithm

Keld Helsgaun, Roskilde University, December 2016

LKH provides a genetic algorithm in which a population of tours evolves toward a population of better tours. The evolution starts from a population of randomly generated tours and repeatedly modifies the population.

- (1) At each step, the algorithm selects two tours from the current population and uses them as parents to produce a child tour.
- (2) The child is subject to local search.
- (3) If the improved child is better than some of the tours of the population, then it replaces one of those tours.

The implementation in LKH of this hybrid steady-state genetic algorithm may be described by the following pseudocode:

```
1. P = {};  
2. INITIAL_TOUR = [];  
3. for (run = 1; run < RUNS; run++) {  
4.     T = ILK(INITIAL_TOUR, MAX_TRIALS);  
5.     T = ITP(T, P);  
6.     if (!has_cost(T, P) {  
7.         if (|P| < POPULATION_SIZE)  
8.             P = P ∪ {T};  
9.         else {  
10.            if (cost(T) < worst_cost(P))  
11.                REPLACE(T, P);  
12.            (T1, T2) = SELECT(P);  
13.            INITIAL_TOUR = CROSSOVER(T1, T2);  
14.        }  
15.    }  
16. }
```

Comments:

Line 1: The algorithm starts with an empty population, P.

Line 2: INITIAL\_TOUR denotes the starting tour for local search (ILK). If it is empty, the starting tour is randomly generated. When a population of a desired size, POPULATION\_SIZE, has been obtained, then INITIAL\_TOUR is the current child tour.

Line 3: The number of steps in the genetic algorithm is specified by the parameter RUNS.

Line 4: INITIAL\_TOUR is improved using LKH's Iterated Lin-Kernighan algorithm (ILK). The maximum number of iterations is given by the parameter MAX\_TRIALS.

Line 5: The tour  $T$  is subject to Iterative Partial Transcription (ITP) with each of the tours in the current population. ITP is a general procedure for improving the performance of a local search based heuristic algorithm. It attempts to improve two individual solutions by replacing certain parts of either solution by the related parts of the other solution. The procedure may be applied to the TSP by searching for sub-chains of two tours, which contain the same cities in a different order and have the same initial and final cities.

Line 6: To prevent premature convergence, a tour  $T$  is only added to the population if no tour of the population has the same cost as  $T$ .

Lines 7-8: When the size of the population is less than `POPULATION_SIZE`, then  $T$  is merely added to the population.

Lines 10-11: If  $T$ 's cost is less than the cost of the worst tour in the population, then  $T$  replaces one of the tours in the population, not necessarily the worst. LKH implements a replacement strategy called RD/CW [2]. The idea is to replace a tour in the population, with a higher cost and with a lower contribution to diversity, by  $T$ .

Line 12: Two parent tours,  $T_1$  and  $T_2$ , are selected with random linear bias towards the best tours of the population.

Line 13: The CROSSOVER operator generates a child tour from two parent tours using Edge Recombination, ER [3]. This operator attempts to preserve the edges of the parents in order to pass on a maximum amount of information to the child. The implementation in LKH is based an extension of ER, called ERX-T [4], which encourages the alternation of parents in edge inheritance with the aim of increasing diversity. The implementation does not include the greedy operations of ERX-T.

## **Parallelization**

The described genetic algorithm may easily be embedded into a parallel distributed client/server application. The server stores the current population, and performs replacement, selection and crossover. The clients perform local search on initial tours received from the server and sends the improved tours back to the server. When a client receives a task, it creates a parameter file and passes it to LKH. The current implementation uses GNU sockets for the communication between server and clients.

## References

- [1] A. Möbius, B. Freisleben, P. Merz, and M. Schreiber,  
*Combinatorial Optimization by Iterative Partial Transcription*.  
Phys. Rev. E 59(4), 4667-4674, 1999.
  
- [2] M. Lozano, F. Herrera, and J. R. Cano.  
*Replacement strategies to preserve useful diversity in steady-state genetic algorithms*.  
Inf. Sci. 178, 23, 4421-4433, 2008.
  
- [3] L. D. Whitley, T. Starkweather, and D. Fuquay,  
*Scheduling Problems and Traveling Salesmen: The Genetic Edge Recombination Operator*.  
International Conference on Genetic Algorithms, 133–140, 1989.
  
- [4] C.-K. Ting,  
*Improving edge recombination through alternate inheritance and greedy manner*.  
Evo COP 2004, 210–219, 2004.