Using Adaptive Operators in Genetic Search

Jonatan Gómez[1], Dipankar Dasgupta[2], and Fabio González[1]

[1] Division of Computer Science, The University of Memphis, Memphis TN 38152
and Universidad Nacional de Colombia, Bogotá, Colombia
{jgomez,fgonzalz}@memphis.edu
[2] Division of Computer Science, The University of Memphis, Memphis TN 38152
dasgupta@memphis.edu

Abstract. In this paper, we provided an extension of our previous work on adaptive genetic algorithm [1]. Each individual encodes the probability (rate) of its genetic operators. In every generation, each individual is modified by only one operator. This operator is selected according to its encoded rates. The rates are updated according to the performance achieved by the offspring (compared to its parents) and a random learning rate. The proposed approach is augmented with a simple transposition operator and tested on a number of benchmark functions.

1 Simplified Hybrid Adaptive Evolutionary Algorithm (SHA-EA)

Parameter Adaptation (**PA**) eliminates the parameter setting of evolutionary algorithms (**EA**s) by adapting those through the execution of the EA [2,3]. Several PA techniques have been developed [2,4,5]. We introduced an hybrid technique for adapting genetic operator probabilities in our previous work [1]. The operator probabilities along with a learning rule rate, which determines the reward/punishment factor on the operator rate, were encoded in each individual. In this paper, we removed the encoding of the learning rate and simulated it with a uniform random rate [0,1]. Also, we used standard operators that only modify the solution part. When a non-unary operator is selected, the additional parents are chosen with a local selection strategy. The operator rates are evolved according to the performance achieved by the offspring (compared to its parent) and the random learning rate generated, according to the Algorithm 1. We tested several functions using different sets of operators. We used elitist selection as replacement strategy, with each EA run for 1000 iterations, and reported the results averaging 50 run. We used a population of 100 individual for binary functions and 200 for real valued functions, using 32 bits for encoding each real value. Table 1, compares the results obtained by SHA-EA with some reported results in the literature.

2 Conclusions

We presented a simplified version of the adaptive evolutionary algorithm proposed by Gomez and Dasgupta [1]. Experimental results showed that the proposed EA was able to determine the usefulness of some operators finding the solution. The performance of the proposed EA is, in general, better than several EAs with well tuned parameters or with the parameter adaptation strategies found in the literature [2,6,7].

E. Cantú-Paz et al. (Eds.): GECCO 2003, LNCS 2724, pp. 1580–1581, 2003.

Algorithm 1 Simplified Hybrid Adaptive Evolutionary Algorithm (SHA-EA)

```
SHA-EA( λ, termination_condition )
 P_0 = initPopulation( λ ) ,
 t_0 = 0
 while( termination_condition( t, P_t ) is false )
  P_{t+1} = GENERATEPOPULATION( P_* )
  t = t+1

GENERATEPOPULATION( P, operators )
 P' = {}
 for each ind ∈ P
  rates = extract_rates( ind )
  δ = random(0,1)    // learning rate
  oper = select( operators, rates )
  parents = select( arity(oper)-1, P, ind ) ∪ {ind}
  ind' = apply( oper, parents )
  if( fitness( ind' ) > fitness( ind ) ) then
   rates[oper] = (1.0 + δ)*rates[oper] //reward
  else  rates[oper] = (1.0 - δ)*rates[oper] //punish
  normalize_rates( ind' )
  P' = P' + select{ ind, ind' }
 return P'
```

Table 1. Best solutions found with different strategies. Third row shows the SHA-EA using mutation (M) and crossover (C) operations, but without using simple transposition (T) operation

EA	RoyalRoad	Deceptive-3	Rosenbrock	Schwefel	Rastrigin	Griewangk
M-SHA-EA	19.04	292.64	0.00012756	366.005	8.351	0.0480
MC-SHA-EA	53.44	298.16	0.01060971	584.553	2.640	0.3212
MT-SHA-EA	20.00	292.08	0.00001939	120.645	0.589	0.1155
MCT-SHA-EA	64.00	300.00	0.00314993	29.741	0.000	0.0008
Tuson [2]	40.64	289.68	-	-	-	-
Digalakis [6]	-	-	0.40000000	-	10.000	0.7000
Patton [7]	-	-	-	-	4.897	0.0043

References

1. J. Gomez and D. Dasgupta, "Using competitive operators and a local selection scheme in genetic search," in *Late-breaking papers GECCO 2002*, 2002.
2. A. Tuson and P. Ross, "Adapting operator settings in genetic algorithms," *Evolutionary Computation*, 1998.
3. F. Lobo, *The parameter-less genetic algorithm: rational and automated parameter selection for simplified genetic algorithm operation.* PhD thesis, Nova University of Lisboa, 2000.
4. L. Davis, "Adapting operator probabilities in genetic algorithms," in *Third International Conference on Genetic Algorithms and their Applications*, pp. 61–69, 1989.
5. B. Julstrom, "What have you done for me lately? adapting operator probabilities in a steady-state genetic algorithm," in *Sixth International Conference on Genetic Algorithms*, pp. 81–87, 1995.
6. J. Digalakis and K. Margaritis, "An experimental study of benchmarking functions for genetic algorithms," in *IEEE Conferences Transactions, Systems, Man and Cybernetics*, vol. 5, pp. 3810–3815, 2000.
7. A. Patton, T. Dexter, E. Goodman, and W. Punch, "On the application of cohort-driven operators to continuous optimization problems using evolutionary computation," *Evolutionary Programming*, no. 98, 1998.