An Incremental and Non-generational Coevolutionary Algorithm

Ramón Alfonso Palacios-Durazo[1] and Manuel Valenzuela-Rendón[2]

[1] Lumina Software, apd@luminasoftware.com
http://www.luminasoftware.com/apd
Washington 2825 Pte C.P. 64040, Monterrey N.L., Mexico
[2] ITESM, Monterrey
Centro de Sistemas Inteligentes
valenzuela@itesm.mx,
http://www-csi.mty.itesm.mx/~mvalenzu
C.P. 64849 Monterrey, N.L., Mexico

The central idea of coevolution lies in the fact that the fitness of an individual depends on its performance against the current individuals of the opponent population. However, coevolution has been shown to have problems [2,5]. Methods and techniques have been proposed to compensate the flaws in the general concept of coevolution [2]. In this article we propose a different approach to implementing coevolution, called *incremental coevolutionary algorithm* (ICA) in which some of these problems are solved by design. In ICA, the importance of the coexistance of individuals in the same population is as important as the individuals in the opponent population. This is similar to the problem faced by learning classifier systems (LCSs) [1,4]. We take ideas from these algorithms and put them into ICA.

In a coevolutionary algorithm, the fitness landscape depends on the opponent population, therefore it changes every generation. The individuals selected for reproduction are those more promising to perform better against the fitness landscape represented by the opponent population. However, if the complete population of parasites and hosts are recreated in every generation, the offspring of each new generation face a fitness landscape unlike the one they where bred to defeat. Clearly, a generational approach to coevolution can be too disruptive. Since the fitness landscape changes every generation, it also makes sense to incrementally adjust the fitness of individuals in each one. These two ideas define the main approach of the ICA: the use of a non-generational genetic algorithm and the incremental adjustment of the fitness estimation of an individual. The formal definition of ICA can be seen in figure 1.

ICA has some interesting properties. First of all, it is not generational. Each new individual faces a similar fitness landscape than its parents. The fitness landscape changes gradually, allowing an arms race to occur. Since opponents are chosen proportional to their fitness, an individual has a greater chance of facing good opponents. If a particular strength is found in a population, individuals that have it will propagate and will have a greater probability of coming into competition (both because more individuals carry the strength, and because a

E. Cantú-Paz et al. (Eds.): GECCO 2003, LNCS 2723, pp. 371–372, 2003.

greater fitness produces a higher probability of being selected for competition). If the population overspecializes, another strength will propagate to maintain balance. Thus, a natural sharing occurs.

```
(*Define A(x, f)*): A(x, f) = tanh(x/f)
Generate random host and parasite population
Initialize fitness of all parasites S_p ← M_p/C_p and hosts S_h ← A_s/M_a
repeat
    (* Competition cycle*)
    for c ← 1 to N_c
        Select parasite p and host h proportionally to fitness.
        error ← abs(Result of competition between h and p )
        S_p ← S_p + M_p A(error, E_error) − C_p S_p(t)
        S_h ← S_h + A_s(1 − A(error, E_error)) − M_a S_h
    end-for c
    (* 1 step of a GA*)
        Select two parasite parents (p_1 and p_2) proportionally to S_p
        Create new individual p_0 by doing crossover and mutation
        S_p0 ← (S_p1 + S_p2)/2
        Delete parasite with worst fitness and substitute with p_0
    Repeat above for host population
until termination criteria met
```

Fig. 1. Incremental coevolutionary algorithm

The equations for incrementally adjusting fitness can be proven to be stable doing an analysis similar to the one used for LCSs [3]. ICA was tested finding trigonometric identities and was found to be robust, able to generate specialization niches and to consistently outperform traditional genetic programming.

References

1. John Holland. Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule-based systems. *Machine Learning: An Artificial Intelligence Approach*, 2, 1986.
2. Christopher D. Rosin and Richard K. Belew. Methods for competitive co-evolution: Finding opponents worth beating. In Larry Eshelman, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 373–380, San Francisco, CA, 1995. Morgan Kaufmann.
3. Manuel Valenzuela-Rendón. *Two Analysis Tools to Describe the Operation of Classifier Systems*. PhD thesis, The University of Alabama, Tuscaloosa, Alabama, 1989.
4. Manuel Valenzuela-Rendón and E. Uresti-Charre. A nongenerational genetic algorithm for multiobjective optimization. In *Proceedings of the Seventh International Conference on Genetic Algorithms*, pages 658–665. Morgan Kaufmann, 1997.
5. Richard A. Watson and Jordan B. Pollack. Coevolutionary dynamics in a minimal substrate. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 702–709, San Francisco, California, USA, 7-11 2001. Morgan Kaufmann.