# Solving Similar Problems using Genetic Algorithms and Case-Based Memory

**Sushil J. Louis**
Department of Computer Science
University of Nevada
Reno - 89557
email: sushil@cs.unr.edu
and Judy Johnson
Department of Computer Science
University of Nevada
Reno - 89557
email: sushil@cs.unr.edu

## Abstract

This paper uses genetic algorithms augmented with a case-based memory of past problem solving attempts to obtain better performance over time on sets of similar problems. When confronted with a problem we seed a genetic algorithm's initial population with solutions to similar, previously solved problems and the genetic algorithm then adapts its seeded population toward solving the current problem. We address the issue of selecting "appropriate" cases for injection and introduce a methodology for solving similar problems using genetic algorithms combined with case-based memory. Combinational circuit design serves as a structured testbed and provides insight that is used to validate the feasibility of our approach on other problems. Results indicate that seeding a small percentage of the population with "appropriate" cases improves performance on similar problems and that the combined system usually takes *less* time to provide a solution to a new problem as it gains experience (memory) from solving other similar problems.

## 1 INTRODUCTION

Genetic algorithms (GAs) are randomized parallel search algorithms that search from a population of points (Holland, 1975; Goldberg, 1989). We typically randomly initialize the starting population so that a genetic algorithm can proceed from an unbiased sample of the search space. However, we often confront sets of similar problems. It makes little sense to start a problem solving search attempt from scratch with a random initial population when previous search attempts may have yielded useful information about the search space. Instead, seeding a genetic algorithm's initial population with solutions to similar previously solved problems can provide information (a search bias) that can reduce the time taken to find a quality solution. Our approach borrows ideas from case-based reasoning (CBR) in which old problem and solution information, stored as cases in a case-base, help solve a new problem (Riesbeck and Schank, 1989).

Although in this paper we report on work using genetic algorithms and a case-base of past experience, our approach is not limited to either genetic algorithms or to case-based memory. In general, combining a robust search algorithm with some implementation of an associative memory can result in a robust learning system that learns, with experience, to solve similar problems quickly. Our system's performance on a test problem class (reported in section 6) supports this view for genetic algorithms and case-based memory.

The next section describes our system and discusses previous work. We provide short descriptions of our experience with problems from combinatorial optimization, structure design, and robot navigation to highlight important issues and establish the feasibility of combining genetic algorithms with a case-based memory. Combinational circuit design is used to show that injecting partial solutions can result in better performance (fitness versus time) (Liu, 1996). The open shop scheduling and re-scheduling problem validates our methodology (Xu and Louis, 1996) and the traveling salesperson problem deals with using information from similar problems of different sizes (Louis and Li, 1997a). Robot navigation adds an important step to the developed methodology when we want to combine solutions to sub-problems to design a solution to a more complex problem (Louis and Li, 1997b). The results indicate that adding a case-based memory improves performance if the right number of appropriate