TIME-SAVINGS TECHNIQUES
TIME-SAVING TECHNIQUES FOR GA AND GP

• Avoid re-computation of fitness for the reproduction operation (which may be performed, say, 9% of the time). This can be used only when the fitness cases do not vary from generation to generation.
• Consider reducing accuracy of fitness calculation. Consider granularity of the data actually required to adequately solve the problem (e.g., the 12 sonar distances to the nearest walls)
• Consider reducing the precision of the fitness calculation. For example, a short float data type may be available on your computer.
TIME-SAVING TECHNIQUES — CONTINUED

• Consider reducing the number of fitness cases
  • Fewer fitness cases often work well
  • Potentially increase number of fitness cases as run progresses
• "Rational Allocation of Trials" (RATS): In this approach, one only executes as many fitness cases as necessary to distinguish individuals (beyond a statistical doubt). This concept appears in the literature of many techniques of machine learning.
TIME-SAVING TECHNIQUES — CONTINUED

• Use table look-ups when either the state transition equations of the system being simulated or the function set involves a lot of time-consuming (e.g., transcendental) functions. This was used in wall follower and box mover problems in GP-1 book.

• Look-up table can be created and used in lieu of direct function evaluation (e.g., Boolean problems, cellular automata problems)
  • Especially useful if there are large number of points in individual program
  • Especially useful with ADFs
  • Especially useful with large number of fitness cases
TIME-SAVING TECHNIQUES — CONTINUED

• Terminate simulations when
  • the values of all the state variables of the system stabilize
  • a trajectory through the state space of the problem is unacceptable for a reason exogenous to the mathematical calculation (e.g., the broom swings through the earth)
  • they just take too long. Often, a small percentage of the individuals consume an inordinate percentage of the computer time.

• Put a cap time spent by iterative operator on one iteration and on individual as a whole for any particular fitness case.
TIME-SAVING TECHNIQUES — CONTINUED

- Consider unobvious parts of the algorithm, such as the randomizer
- Consider other parts of the algorithm, such as the roulette wheel (possibly replacing it with an indexing scheme, sort, fast sort, or tournament selection)
- Disable graphics and other instrumentation for in production runs
- Use the metering and profiling software!!!
TIME-SAVING TECHNIQUES — CONTINUED

• Parallel computers
• Rapidly reconfigurable field programmable gate arrays (FPGA) to get massive parallelization at hardware speeds or field programmable transistor arrays (FPTA) for analog circuits containing transistors
ACCELERATING GP

• If the number of fitness cases to be handled by each individual in the population is large, consider compiling (in GP) each individual program
• GP assembly code (Peter Nordin)
MEMORY-SAVING TECHNIQUES

• 1-byte representation for GP
  • Permits about 200 random constants (and about 56 functions)
  • Also: 2-byte representation

PROGRAMMER-SAVING TECHNIQUES

• Optimize your time, too. Many parts of the code are virtually irrelevant to optimization
COMMON MISTAKES IN APPLYING GENETIC ALGORITHMS

- Population is MUCH TOO small
- Mutation rate is TOO HIGH
- Excessive GREED is introduced
- Improper initialization
- Fitness is not adequately gradated
- Hand-crafted crossover operators cause mutation to be introduced for virtually every crossover
- Hand-crafted crossover operators are not in sync with the problem
- Misapplying rules of thumb