Problem Set No. 4
BMI 226 / CS 426
Fall 2003

NAME_____

USING GENETIC PROGRAMMING SOFTWARE

PLEASE ATTACH A COPY OF THE PROBLEM-SPECIFIC CODE YOU WRITE AND THE OUTPUT OF YOUR COMPUTER RUN

This problem set may be done with "Dave's Genetic Programming Code in C" (DGPC) (by David Andre), LilGP (by Bill Punch), or ECJ (by Sean Luke) software. The references below are to variable names used in DGPC, but the names are self-explanatory.

PROBLEM 1: SYMBOLIC REGRESSION OF $X^2/2 + 2X + 2$

Use a function set consisting of the addition (+), subtraction (-), multiplication (*), and protected division (%) to do a symbolic regression of the target function $X^2/2 + 2X + 2$. The protected division function % is protected against division by zero (page 82, *GP* book). Do not include random constants in the terminal set. Use a population of size of M = 1,000. Use Maximum number of generations to be run, G, of 151. Set MaxNewTreeDepth to 5, crossover_fraction_for_leaves of 0.10, crossover_fraction_for_node of 0.80, copy_fraction of 0.10, and mutation_fraction of 0.00.

QUESTIONS:

(1a) Symbolic regression of $X^2/2 + 2X + 2$:.Fill in the following tableau for this problem.

Tableau for symbolic regression of $X^2/2 + 2X + 2$

Objective:	
Terminal set:	
Function set:	
Fitness cases:	
Raw fitness:	
Standardized fitness:	
Hits:	
Wrapper:	
Parameters:	*M* = _____ . *G* = _____.

Success predicate:	

(1b) Generation 0 performance: What is the best-of-generation individual of generation 0 and its fitness value?

(1c) **IMPORTANT QUESTION:** Pick out some interesting-looking and explainable best-of-generation individual for some intermediate generation that illustrates progress toward the solution. Write out the individual, its generation number, and its fitness. Why this individual has an intermediate value of fitness (i.e., what is there about its structure that gives it intermediate fitness, as opposed to best, as opposed to what you saw at generation 0)?

(1d) **The Best-of-Run Individual:** Write the generation number of the final generation of the run, the best-of-generation individual of this final generation, and its fitness value.

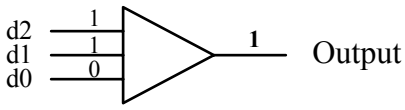
(1e) Was this a perfect (i.e., algebraically correct) or approximate solution to the problem?

(1f) Try running this problem at least more time (or until one run yields an algebraically correct solution). Write the generation number of the final generation of the run, the best-of-generation individual of this final generation, and its fitness value.

(2a) Symbolic Regression of Boolean Even-3-Parity Functions WITHOUT Automatically Defined Functions:

In Boolean function learning, we seek to find the Boolean function that returns particular Boolean output values. That is, we seek to do symbolic regression in the Boolean domain.

The Boolean *even-k-parity function* of *k* Boolean arguments returns `T` (true) if an even number of its Boolean arguments are `T`, and otherwise returns `NIL` (0). Parity functions are often used to check the accuracy of stored or transmitted binary data in computers because a change in the value of any one of its arguments toggles the value of the function. Because of this sensitivity to its inputs, the parity function may be difficult to learn and is therefore often used as a benchmark in the fields of machine learning and neural networks.



Try Boolean function learning with the Boolean even-3-parity problem.

Tableau without ADFs for the even-3-parity problem

Objective:	Find a program that produces the value of the Boolean even-3-parity function as its output when given the value of the three independent Boolean variables as its input.
Terminal set without ADFs:	`D0`, `D1`, and `D2`.
Function set without ADFs:	`AND`, `OR`, `NAND`, and `NOR`.
Fitness cases:	All $2^3 = 8$ combinations of the three Boolean arguments `D0`, `D1`, and `D2`.
Raw fitness:	The number of fitness cases for which the value returned by the program equals the correct value of the even-3-parity function.
Standardized fitness:	The standardized fitness of a program is the sum, over the $2^3 = 8$ fitness cases, of the Hamming distance (error) between the value returned by the program and the correct value of the Boolean even-3-parity function.
Hits:	Same as raw fitness.
Wrapper:	None.
Parameters:	$M = 1{,}000$. $G = 51$.
Success predicate:	A program scores the maximum number (i.e., 8) of hits.

(2a) For generation 0 of one run, what was the best-of-generation individual in the population?

(2b) What fitness did it have?

(2c) What answer does this best-of-generation individual from generation 0 produce for the 8 fitness cases for this problem? That is, fill in the last column of this truth table for your best-of-generation 0 individual.

Fitness case	**D2**	**D1**	**D0**	**Even-3-parity**	**Your best-of-generation 0 individual**
0	NIL	NIL	NIL	T	
1	NIL	NIL	T	NIL	
2	NIL	T	NIL	NIL	
3	NIL	T	T	T	
4	T	NIL	NIL	NIL	
5	T	NIL	T	T	
6	T	T	NIL	T	
7	T	T	T	NIL	

(2d) Based on this truth table, what Boolean function is performed by your best-of-generation 0 individual?

(2e) Write the generation number of the final generation of the run, the best-of-generation individual of this final generation, and its fitness value.

(3a) Symbolic Regression of Boolean Even-3-Parity Functions WITH Automatically Defined Functions:

Tableau with ADFs for the even-3-parity problem

Architecture of the overall program with ADFs:	One result-producing branch and one two-argument function-defining branch, ADF0.
Parameters:	Branch typing.
Terminal set for the result-producing branch:	D0, D1, and D2.
Function set for the result-producing branch:	**ADF0**, AND, OR, NAND, and NOR.
Terminal set for the function-defining branch ADF0:	ARG0 and ARG1.
Function set for the function-defining branch ADF0:	AND, OR, NAND, and NOR.

QUESTIONS:

(3a) For generation 0 of one run, what was the best-of-generation individual in the population?

(3b) What fitness did it have?

(3c) What answer does this best-of-generation individual from generation 0 produce for the 8 fitness cases for this problem? That is, fill in the last column of this truth table for your best-of-generation 0 individual.

Fitness case	**D2**	**D1**	**D0**	**Even-3-parity**	**Your best-of-generation 0 individual**
0	NIL	NIL	NIL	T	
1	NIL	NIL	T	NIL	
2	NIL	T	NIL	NIL	
3	NIL	T	T	T	
4	T	NIL	NIL	NIL	
5	T	NIL	T	T	
6	T	T	NIL	T	
7	T	T	T	NIL	

(3d) Fill in the last column of this truth table for the ADF0 of the best-of-generation individual of generation 0?

Fitness case	**ARG1**	**ARG0**	**ADF0 from your best-of-generation 0 individual**
0	**NIL**	**NIL**	
1	**NIL**	**T**	
2	**T**	**NIL**	
3	**T**	**T**	

(3e) Based on this truth table, what Boolean function is performed by your ADF0?

(3f) Was this ADF0 actually called from the result-producing branch, RPB?

(3g) Write the generation number of the final generation of the run, the best-of-generation individual of this final generation, and its fitness value.

(3h) If the result-producing branch of your run didn't refer to its ADF0, try running this problem several more times until you get a run that solves the problem and that actually uses ADF0. Write the generation number of the final generation of the run, its fitness value, and the best-of-generation individual of this final generation (or attach a copy of it if it's really big).

(3i) For your solution above that actually uses ADF0, fill in the last column of this truth table for your ADF0.

Fitness case	ARG 1	ARG 0	ADF0 from your best-of-run individual
0	NIL	NIL	
1	NIL	T	
2	T	NIL	
3	T	T	

(3j) Based on this truth table, what Boolean function is performed by your ADF0?

(3k) Compare the number of fitness evaluations (i.e., number of generations times the population size) for a successful run of this problem WITHOUT ADFs versus a successful run of this problem WITH ADFs. Which run solved faster?

APPENDIX

Truth table for the three-argument Boolean rule 235 for (or (even-2-parity D2 D1) D0)

	D2	D1	D0	Rule 235
0	NIL	NIL	NIL	T
1	NIL	NIL	T	T
2	NIL	T	NIL	NIL
3	NIL	T	T	T
4	T	NIL	NIL	NIL
5	T	NIL	T	T
6	T	T	NIL	T
7	T	T	T	T