Exam Evolutionary Computing
23.10.2013
ANSWERS

1. There are two options.

   (a) Use a polynomial function for approximation and optimize the coefficients using evolution. That means using an Evolution Strategy for continuous optmization.

      i. **(5 pt)** The genotype is a string of real numbers. The length depends on the number of factors N and the order of the polynomial.

      ii. **(5 pt)** We use the model phenotype (polynomial with the coefficients of the candidate solution being evaluated) on all the examples provided in the history data. If p_i is the number of tickets the model predicts and t_i is the number of tickets actually sold for each example i, then the fitness is calculated as $\sum (p_i - t_i)^2$

      iii. **(2 pt)** Averaging crossover. (*or you could write any discrete recombination (book 3.5.1) or recombination for real numbers (book 3.5.3))

      iv. **(2 pt)** Gaussian mutation. (* or uniform mutation (book 3.4.3))

      v. **(2 pt)** Tournament. (or you could pick any other parent selection (book 3.7))

      vi. **(2 pt)** Tournament for (μ, λ) (*or any other selection (book 3.8))

      vii. **(2 pt)** Random otherwise motivate and explain.

      viii. **(2 pt)** Evaluations limit or generations with no improvement limit.

   (b) Use Genetic Programming to evolve a full model from scratch.

      i. **(5 pt)** The genotype is a syntactic tree; it can represent a full procedural program or just a mathematical formula. In any case we must define a function set and a terminal set. We'll give an example taking the simpler case of evolving mathematical formulas. Our function set could be $\{+, -, \cdot, \div, cos, sin, pow, exp, log, ln\}$. The terminal set would be $\Re \cup \{v_1, ..., v_N\}$. It is important that the input variables are always included in the terminal set.

      ii. **(5 pt)** We use the model phenotype (formula or program) on all the examples provided in the history data. If p_i is the number of tickets the model predicts and t_i is the number of tickets actually sold for each example i, then the fitness is calculated as $\sum (p_i - t_i)^2$

      iii. **(2 pt)** Subtree crossover.

      iv. **(2 pt)** Replace subtree (with random new subtree).

      v. **(2 pt)** Fitness proportional with roulette wheel. (or you could pick any other parent selection (book 3.7))

      vi. **(2 pt)** Generational. (*or any other survivor selection (book 3.8))

      vii. **(2 pt)** Ramped half-and-half (book 6.8).

      viii. **(2 pt)** Evaluations limit or generations with no improvement limit.

2. **(15 pt)** (book 13.3) The first main difficulty with evolving art and music is defining an appropriate representation. Phenotypes are very complex (visual artwork or a music track)

and we also need to facilitate meaningful crossover and mutation operators. It is necessary to define a component-based representation where the genotypic components are decoded (through growth/embriogeny) into a collection of application-specific objects that compose the phenotype. The second problem is how to evaluate individuals since artistic value is subjective and difficult to encode as a mathematical formula. A way around this problem is to use interactive evolution where a human mannualy performs the fitness evaluation or the selection process. This of course makes the evolutionary process very slow and noisy (due to human inconsistencies and fatigue).

3. (a) **(3 pt)** (book 11.4) Yes. There are a finite number of states (all possible binary configurations of the individuals in the population) for the system. Furthermore, the next population depends only on the current population (though it cannot be predicted due to the stochastic nature of the variation operators, thus, the probability for a next state is solely dependent on the current state.

   (b) **(4 pt)** (book 11.6) The takeover time is the number of generations needed for the fittest individual to take over the whole population in the absence of variation operators. The mixing time is the time needed by recombination to bring together the building blocks of the optimal solution. The mixing time should be less than the takeover time so that all possible combinations of building blocks can be tested before a fit (but not optimal) individual takes over the population.

4. (a) (book 12.2 - 12.4)
      i. **(2 pt)** CSP
      ii. **(2 pt)** COP
      iii. **(2 pt)** COP
      iv. **(2 pt)** CSP

   (b) **(5 pt)** The constraint of case (ii) can be handled with repair. You can go through the investments defined in an individual until the limit of funds is reached and delete the rest. The constraints of (iii) can also be handled with repair. Given an ordering of operations, if an operation is to be executed before its prerequisites are met then it is put aside and it is later inserted as soon as it is possible to perform it.

   (c) **(5 pt)** Case (ii) can be handled by maintaining feasibility. The representation can be a string of numbers, one for each company. Initialization can randomly split the available funds into k numbers and assign these to randomly selected companies. Mutation will subtract a random amount from a company (making sure to not result in a negative investment) and then add this amount to another company randomly. It can be easily shown that using averaging crossover between two feasible solutions will create a feasible solution for this problem. Also, some of the constraints of case (iv) can be handled by maintaining feasibility. In specific, the requirement that "no two queens check each other" translates into three constraints: no two queens can be in the same row, column or diagonal. The first two can be taken care of by using a permutation of 8 numbers so that value i_n denotes that for column n a queen is placed at row i_n (book 2.4.1). Permutation specific operators can then be used.

   (d) **(2 pt)** The constraints of case (i) must be handled by penalties. We obviously cannot preserve feasibility or repair solutions because if we had any feasible solution we would

have already solved the problem. From another perspective this CSP problem does not have an objective function of its own to use as a fitness function for our EA. Thus, we need to use penalties as a fitness function, e.g. assign an individual as a fitness the number of conjunctions in the logical expression that are not true making this a minimization problem. Similar, we need penalties for case (iv) even though some of the constraints (two queens cannot be on the same row or column) can be taken care of by proper representation as explained above. The final constraint of this CSP must be handled with penalties because we cannot repair or maintain feasibility and because we need a fitness function.

5. **(5 pt)** Since no opponent or training examples are available the solution is to use competitive co-evolution (book 13.2.2). That means that the fitness of individuals is indirectly decided by having them play againist each other. An efficient way to handle selection is to use binary tournament selection (book 3.7.4). Whenever a parent/survivor must be selected, two individuals can be randomly taken from the pool and the winner decided by having them play against each other.

6. (a) **(5 pt)** You can choose from multiple populations (book 9.3.1), spatial distribution (book 9.3.2), fitness sharing (book 9.4.1) and crowding (book 9.4.2).

   (b) **(6 pt)** We first calculate the Hamming distances between all parent-child pairs by counting the number of different bits:
   $d(A,C) = 2 \;\; d(A,D) = 1 \;\; d(B,C) = 3 \;\; d(B,D) = 4$
   Now, according to the process of deterministic crowding (book 9.4.2), we must pair parents and children so that the distances between pairs are minimized. We can easily see that this is accomplished by pairing A with D and B with C. Thus, $p_1 = A$, $p_2 = B$, $o_1 = D$ and $o_2 = C$. We can confirm the inequality
   $d(p_1,o_1)+d(p_2,o_2) = d(A,D)+d(B,C) = 1+3 = 4 < 6 = 2+4 = d(A,C)+d(B,D) = d(p_1,o_2) + d(p_2,o_1)$

   (c) **(2 pt)** There are genes for which both children have the value of parent A, thus, no N-point crossover could have been used. The only choice left is uniform crossover.

7. **(8 pt)** Both in favor and against answers can be valid if backed up by proper arguments. In favor: . Against: .