Exam Evolutionary Computing
11.01.2011

NOTES:

1. **YOUR NAME MUST BE WRITTEN ON EACH SHEET IN CAPITALS.**
2. You can answer the questions in English or in Dutch.
3. This is an 'open book' exam. You can use the course book – but nothing else.
4. Points to be collected: 90, free gift: 10 points, maximum total: 100 points.
5. Grade: total number of points divided by 10.

QUESTIONS

1. We are to solve a graph 3-coloring problem with evolutionary computing. That is, we have a graph $G = (N, E)$ with $n = |N|$ nodes and $m = |E|$ edges and three colors $\{r, w, b\}$. We define a coloring as an assignment of colors to all nodes. Then the task is to find a coloring such that no neighboring nodes have the same color.
(2p) What kind of problem is this, an FOP, a COP, or a CSP?

2. We decide to represent a coloring by a vector $x = \langle x_1, \ldots, x_n \rangle \in \{r, w, b\}^n$, where the k-th position belongs to node $k \in N$ and x_k is the color of k. Constraints are denoted as $\{c_1, \ldots, c_m\}$. For each edge $e = (k, l) \in E$ there is a unique constraint c_i such that $c_i(x) = \mathit{true}$ if and only if $x_k \neq x_l$. Furthermore, we use the notation C^k for the set of constraints involving variable x_k (that is, involving the node k). Now we can define two different fitness functions as follows:

   $f_1(x) = \sum_{i=1}^{m} A(x, c_i)$ where

   $$A(x, c_i) = \begin{cases} 1 & \text{if } c_i(x) = \mathit{false} \text{ (i.e., } x \text{ violates } c_i) \\ 0 & \text{otherwise} \end{cases}$$

   and

   $f_2(x) = \sum_{j=1}^{n} B(x, C^j)$ where

   $$B(x, C^j) = \begin{cases} 1 & \text{if } x \text{ violates at least one } c \in C^j \\ 0 & \text{otherwise} \end{cases}$$

   (a) **(5p)** What does the fitness function f_1 measure in terms of the (colored) graph?

   (b) **(5p)** What does the fitness function f_2 measure in terms of the (colored) graph?

   (c) **(6p)** Which of these fitness functions is preferable if we want to use a heuristic mutation operator that 'fixes' some errors in a given chromosomes? Give arguments why.

3. Using the above representation and either fitness functions specify an EA suitable[1] for solving the above problem. In particular, give
   (a) **(3p)** an appropriate crossover operator,
   (b) **(3p)** an appropriate mutation operator,
   (c) **(3p)** an appropriate selection mechanism,
   (d) **(2p)** an initialization method,
   (e) **(2p)** a stop condition,

4. Invent a multi-parent recombination mechanism for permutation representation, such that it is not just a concatenation of a number of two-parent recombination operators. That is, describe a recombination mechanism that can be applied to an arbitrary number of $n > 1$ parents and has the property that if all parents are permutations (over the same alphabet) then so are the offspring. You can solve this problem in two steps:
   (a) **(7p)** Describe a recombination mechanism that can be applied to $n = 3$ parents and permutations over the alphabet $\{a, b, c, d, e, f\}$. Illustrate its working with a concrete example.
   (b) **(10p)** Describe a recombination mechanism for permutations that can be applied to an arbitrary number of $n > 1$ parents and provide an argument to "prove" that it always produces correct offspring. NB. The quotes in "prove" indicate that you needn't provide a formal proof with mathematical rigor.

5. (a) **(3p)** Explain what deterministic, adaptive, and self-adaptive parameter control mean in evolutionary computing.
   (b) **(5p)** Invent a deterministic mechanism to modify the tournament size of a GA over time. Motivate your method by (intuitive) arguments: why would it be helpful?
   (c) **(5p)** Invent an adaptive mechanism to modify the tournament size of a GA over time. Motivate your method by (intuitive) arguments: why would it be helpful?
   (d) **(7p)** Invent a self-adaptive mechanism to modify the tournament size of a GA over time. Motivate your method by (intuitive) arguments: why would it be helpful?

6. **(8p)** What is the difference between uniform crossover and global discrete recombination?

7. **(8p)** Is edge recombination applicable for any problem represented by permutations? Or can it only be used for TSP-like problems?

8. **(6p)** Consider the following statement:

   Evolution Strategies do not suffer from bloat, because they are self-adapting the mutation stepsize.

   Is this statement correct or not? Give arguments.

---

[1] The EA does not have to be "smart" (efficient). But the representation and the operators should be such that a solution can be found.