Biologically Inspired Computing: Assignment, year 2010

Key Details:

Value: This assignment is worth 25% of the module mark

Deadline: Hand it in on or before Friday 19th March 2010

What to hand in: (also see final page): A report with AT MOST 2,000 words, and in which any figures and tables take up AT MOST 4 sides of A4.

How to hand in: Handin **must happen like this**: email a PDF of your assignment report as an attachment, sent to dwcorne@gmail.com
In the body of the message, give me your name and the title of your degree.
The subject line must be exactly as follows: BIC CW1

Testing Various Evolutionary Algorithms on a Constrained Bin-Packing Problem

Here is the problem:
There are a set of *N* items, and each item has a given weight. Also, each item has a given *type* (there are *T* different types of item). The items have to be arranged into *C* containers, in such a way that the total weight of each container is as similar as possible. However there are constraints involving the types. In this assignment, there will always be 5 types, and the constraints are: items of type *i* and *i*+1 (modulo 5) cannot be in the same container.

For example, suppose we have ten items to pack into three containers with weights and types as follows:

Items	1	2	3	4	5	6	7	8	9	10
Type	1	1	2	2	3	3	4	4	5	5
Weights	30	35	60	5	100	110	20	40	90	65

Using the simple encoding described in the lectures, you should understand how the following random individual translates into a solution:

3,2,2,1,3,2,3,1,1,3

This corresponds to a packing as follows:

(container1: 4, 8, 9) (container2: 2, 3, 6) (container3: 1, 5, 7, 10)

Without type constraints, fitness would be:

heaviest_container minus **lightest_container** = 215 – 135 – 80

However, in this case the solution contains the following **six** pairs of invalidly placed items: 8 and 9 (types 4 and 5), 2 and 3 (types 1 and 2), 3 and 6 (types 2 and 3), 10 and 1 (types 5 and 1), 5 and 7 (types 3 and 4), 7 and 10 (types 4 and 5). Therefore, we will penalise it by adding a value to the fitness, *TW*, for each of the invalid pairs of items. In this case, there are 6 invalidly places pairs, so fitness becomes 80 + (6 * *TW*). In this assignment, always set *TW* to 10.

What you have to do

Get the data here: http://www.macs.hw.ac.uk/~dwcorne/Teaching/data.txt
Column 1 just gives an item ID, column 2 gives that item's weight, and column 3 gives its type. (note that the constraint described previously means that items of type 0 and 5 cannot be in the same container). These data are for a 500-item problem with 5 types, and they need to be packed into a varied number of containers (you will use different numbers of containers in different experiments). In all cases you want to minimise the fitness function described previously – i.e. the difference between the heaviest and lightest containers, with penalties added for any invalid pair of items.

Set of Experiments A

Implement a steady-state EA using binary tournament selection and replace-worst. In every case below, run this EA for 10,000 evaluations. The result of a single run will be the fitness of the best individual found by this run. The **result of a set** of 10 runs will be a set of 3 numbers: the **best**, **worst**, and **mean** of the individual results of the 10 runs. In these experiments, set the number of containers to be 50.

Do a set of 10 runs for each of the following algorithms: in cases 1—8, the algorithm is the EA described above, but the list below fills in the missing details:

- Algorithm1: single-gene new-allele mutation, population size 10
- Algorithm2: 2-gene new-allele mutation, population size 10
- Algorithm3: 3-gene new-allele mutation, population size 10
- Algorithm4: 20-gene new-allele mutation, population size 10
- Algorithm5: single-gene new-allele mutation, population size 100
- Algorithm6: 2-gene new-allele mutation, population size 100
- Algorithm7: 3-gene new-allele mutation, population size 100
- Algorithm8: 20-gene new-allele mutation, population size 100
- Algorithm9: Hillclimbing, with single-gene new-allele mutation
- Algorithm10: Hillclimbing, with 2-gene new-allele mutation
- Algorithm11: Hillclimbing, with 20-gene new-allele mutation

Using tables and/or graphs of your results, write a justified response to the following questions:

What is the best mutation operator to use on this problem; what is the best overall algorithm? What are your overall observations and what explanations do you suggest for them?

THAT'S NOT ALL – SEE NEXT PAGE

Your answer to question 1, and of course your work that underpins that answer, can attain up to 70% of the marks for the assignment. The remaining 30% can be gained as follows:

Now, using what you decide are the best THREE of algorithms 1—8 in this case, call them B1, B2 and B3 and do the above sets of 10 runs again with these algorithms, but this time the number of containers should be 20.

Using tables and/or graphs of your results, write a justified response to the following question:

Question 2: How and why does the number of containers affect the algorithms' performance on this problem?

What to hand in: I want to see your responses to questions 1 and 2 also) backed up and justified by tables and/or graphs of your results. The report must include (possibly only by being represented as 3 points on a graph, or bars in a bar chart) the **result** from each **set** of 10 runs. The combined answers to your questions must be AT MOST 2,000 words, and figures and tables should take up AT MOST 4 sides of A4.