CS4444: Software Engineering
CS5420: Software Development

Homework 2

Due: Feb. 8, 2015 at midnight to UNT.CS4444@gmail.com

Implement the AETG algorithm to generate combinatorial test suites for 3-way coverage.

In this assignment, you will extend your homework 1 submission to provide 3-way combinational coverage.

1. For this assignment, you are not required to take strings as input. You may simply give every factor and every level a unique numerical identifier. For instance, the input in Table 1 may be represented as shown in Table 2:

Hardware	Operating System	Network Connection	Memory
PC	Windows XP	Dial-up	64MB
Laptop	Linux	DSL	128MB
PDA	Solaris	Cable	256MB

Table 1 - A component based system with four factors that each have three levels

f_0	f_1	f_2	f_3
0	3	6	9
1	4	7	10
2	5	8	11

Table 2 - A component based system with four factors that each have three levels. (Factor f_0 has levels 0, 1, 2; factor f_1 has levels 3, 4, 5; factor f_2 has levels 6, 7, 8; factor f_3 has levels 9, 10, 11).

2. You should prompt the user for the number of factors and levels so that the grader is able to easily grade your program.
3. You may implement the program in your choice of programming language.
4. Your program may end up using too much time or memory for larger inputs, so you will need to be careful in your implementation. The papers do not describe the exact

implementation details for the exact data structures that they use. It is expected that you can effectively make decisions that lead to a fast and efficient implementation.

5. There is randomization in your algorithm. You will run your program 100 times for each input and report the average size and execution time for each input. You should also use 50 candidates as done in the AETG literature.
6. You will report the results of your algorithm for several inputs listed in the table on the next page. Of course, it is possible that your results will slightly vary from their reported results since there is randomization in the algorithm. However, if your results are off by more than 20% for any inputs larger than 3^4, you have a bug in your program.
7. You should strive to implement an efficient solution. To receive credit for this assignment, your algorithm cannot run for longer than 20 minutes to find a solution.
8. You are only required to implement a solution for up to 3-way coverage.

Submission

Part 1 – Implementation and results (100%)

1. **Code** – You should provide your code and if applicable, your makefile. Please zip the files before e-mailing them to our course e-mail address. If there is an "exe" in your email, gmail will not allow the email to come through, so rename it! The TA will email to confirm that he received your assignment. If you do not receive a confirmation within 24 hours, be sure to contact him.
2. **Submit a Document with your Results** – your results should be close to those published in the AETG literature. You should provide output files for each input such that each output file contains the best covering array generated for that input. You will lose 20% credit if you do not submit your output files. You will also complete the following table with the results from your program:

Input	Your best result	Your average result	Your worst result	Average execution time	mAETG
3^{13}					88
2^{10}					18
2^{12}					21
3^{4}					47
$10^{1}9^{1}8^{1}7^{1}6^{1}5^{1}4^{1}3^{1}2^{1}$					928

3. **Submit output files using this format –** The TA will use a program to confirm that all pairs are covered in your solution. You must use this format for your output. The first line is the number of tests in the solution. A blank line follows this. The remaining lines include the solution. The lines below show a partial solution.

```
9

1 4 7 9
```

```
1 5 8 11
2 4 8 10
2 3 7 11
0 5 7 10
2 5 6 9
1 3 6 10
0 4 6 11
0 3 8 9
```

Here is a commented version of the output format, but please do not include comments in your output files. This is for instructional use only:

```
9  // There are 9 tests in my solution

1 4 7 9     // This is test case 1 (i.e., one level for each factor is assigned)
1 5 8 11   // This is test case 2
2 4 8 10
2 3 7 11
0 5 7 10
2 5 6 9
1 3 6 10
0 4 6 11
0 3 8 9
```

4. **Grading of code -** No credit will be given for code that does not compile or if the size of the solutions are not within 20% of the results reported in the AETG literature (with exception to input 3^4). No credit will be given to code that takes longer than 20 minutes to run on any of the above inputs. Your code must be object-oriented and use well-named methods and variables. There will be a 10% penalty if the code is not commented.