BNFO 591
INTRODUCTION TO HIGH PERFORMANCE COMPUTING IN BIOINFORMATICS AND THE LIFE SCIENCES

FORTRAN FINAL PROJECT: ALL POSSIBLE SUBSETS

TARYNN M. WITTEN

ABSTRACT. In this final project you will be completing a number of programming and analysis tasks which represent the culmination of your 15 weeks in class. In particular, you will be calculating all of the possible subsets 2^{35} of a set of 35 numbers that map to 35 ICD disease codes.

1. Overview

Earlier this semester you heard a guest lecture from Dr. Bruce Carnes. He spoke on the subject of mortality modeling and in particular about trying to understand whether or not there were intrinsic (read that as genetic) processes that formed a smallest possible disease subset that could be used to accurately describe population mortality in a given country. Dr. Carnes, over some 20 years of research effort classified the ICD-9/10 codes (`https://en.wikipedia.org/wiki/International_Statistical_Classification_of_Diseases_and_Related_Health_Problems#ICD-9`. The ICD codes are the billing codes that classify, for insurance purposes, what the healthcare visit was about. Dr. Carnes classified the codes into 35 subsets that we will be using for this final project. Your goal is to develop a program, in `FORTRAN`, that will hypothetically write all possible subsets of a set of m numbers to a datafile. You will **NOT** be actually writing all of these sets out to a datafile. However, at some point we may actually wish to do that so your program has to be set up to execute the full write. You will construct your `FORTRAN` program as follows:

2. PART 1: Ordering and Alphabetizing

- The file `ICD10_Codes_Unordered.txt` contains 35 ICD code groupings in no particular order. Write a `FORTRAN` program that reads the dataset in and outputs the data to a new file called `Alphabetical_ICD.dat` in which the code groupings are in alphabetical order. In addition, you must write a number from $1 - 35$ that corresponds to each ICD code grouping in alphabetical order so that we can later map the number in the set to the ICD code. So your output file should have the first column as a number and the second column as an ICD code grouping.
- Hand in a copy of your program and the alphabetized list.

Date: September 27, 2015.

3. PART 2: All Possible Subsets

- Write a program to actually write out what the number 2^{35} is. Make sure that you show all of the numerals. No exponental formatting allowed. You have to show the whole number as if it were an integer.
- Next, you are to write a program that computes all possible 2^{35} subsets of the 35 ICD code groupings using the numbers $1-35$ to represent the alphabetized ICD codes from Part 1 above.
- Your program should write to an output file called `all_possible_subsets.dat`.
- Write only the first **1**00 subsets to a file. You are going to output the following information to the file in the following layout (all in one line).
  (1) The first number should be the subset number. That is, a number from $1-2^{35}$.
  (2) The next set of numbers should be the subset you calculated.
  (3) This is followed by the number -999.
  (4) This is then followed by the numbers that were NOT in the subset you calculated. For example, suppose that your set is the set of 4 numbers $\{1,2,3,4\}$. Then your first line would look like 1 1 2 3 4 $-$ 999 0 because there isn't anything in the other set. So I am letting you have this as a gimme. Your next line, for example, would look like 2 1 2 3 $-$ 999 4 then 3 1 2 4 $-$ 999 3 until you had listed all 2^4 possible subsets. The last one would be 16 0 $-$ 999 1 2 3 4.
  (5) Note that I should be able to read your file with the program in Appendix [1] of this assignment. So, you might want to copy the program and make sure that it will read your data successfully. And you might want to figure out a way to check that. Just because it reads it doesn't mean that it entered the data into the array correctly!
- Create a counter variable that counts the number of subsets that you calculate and make sure that you check that your counter is equal to the value 2^{35} when your program finishes computing the subsets.
- Hand in a copy of your program and the list of 100 subsets. Show that you executed it on both `Compile` and `Stampede`. Use doublesided printing please and only one list is necessary.

WARNING: DO NOT write all 2^{35} subsets to a file on either `Compile` or `Stampede`!!!! You will crash the system and make people **EXTREMELY** angry with you. Write only the first 100 subsets. Danger Will Robinson, Danger.

HINT: I suggest that you start with $4-6$ codes (make a smaller dataset as a test dataset) and actually hand enumerate all of the subsets. Run your program with that and make sure that it gives you all of the subsets before you run the big code please.

4. PART 3: Timing

Once you have your program working, you are to make a set of timing runs without parallelization. You are to time the portion of your code that actually calculates the subsets. You are to do this as follows.

- Store the execution times in a datafile named `no_opt_timing.dat`. You will execute $N = 10$ timing runs in which you calculate how long it took to complete incremental powers of 10^N calculations where $N = 0, 1, 2, \ldots 10$. Each row of your datafile will contain the following N, 10^N and the time t that it took to execute 10^N subset calculations. Note that 10^{10} is the largest power of 10 less than 2^{35}. We figure that out as follows. We want to find the value of N such that $10^N \leq 2^{35}$. So taking the log of both sides of the equation we have that

$$10^N \leq 2^{35} \tag{1}$$

$$\log 10^N \leq \log 2^{35} \tag{2}$$

$$N \log 10 \leq 35 \log 2 \quad \text{however, } \log 10 = 1 \tag{3}$$

$$N \leq 35 \log 2 = 10.54 \tag{4}$$

- When you have completed running your timing runs and built your output datafile, use `Gnuplot` and plot the $\log(time)$ *vs.* N. Do not connect the datapoints. Label your plot with a title, and axis labels having the correct dimensions. Note that you are really plotting a *log vs. log* plot. What do you see in your plot? Discuss the possible relationship
- Bonus points for fitting a regression curve to the data, reporting the parameter values and interpreting the results.
- Hand in a copy of your programs for this part, the datafiles, the plots and show that you executed the program on both `Compile` and `Stampede`.

5. PART 4: Timing with Threads

In this section of the assignment, you are to take your program and you are going to set it up to use threads in an attempt to see if threading can be used to speed up your program.

- You are going to do this by successively doubling the number of threads, $2^m, m = 1, 2, 3, 4$. You will only do this with three calculation sets. Compute $1,000$, $1,000,000$ and 2^{35} loop calculations. So you should have a dataset that has four times for each of three subset calculation numbers.
- For each subset calculation, use `Gnuplot` to plot the $\log(time)$ *vs.* $\log(2^m)$. So you should have three plots, appropriately labeled with headers and axes.
- Discuss your results. Where and when does it make sense to increase threads?
- Hand in a copy of your programs for this part, the datafiles, the plots and show that you executed the program on both `Compile` and `Stampede`.

6. Closing Comments

- Your final project is due on the day of the final examination.
- There will be no late projects accepted as grades have to be computed very quickly after the examination.
- Staple each part separately.
- All four parts must be clipped together with a binder.
- Name, date and page number on every page.
- NO PAPERCLIPS.

- Projects will be available for pickup at my office once grades are submitted.

7. Appendix 1

7.1. **Program to Test Your Output File.** Here is a little program that I wrote to read your homework output file `all_possible_subsets.dat`. If my program cannot read your file, then something is wrong. You should modify my code to make sure that you are accurately reading the file.

```
program read_my_datafile
  integer, parameter = 100 :: nrecords ! how many records are in the file to be read
  integer, parameter = 37 :: nvars ! how many variables are in the line
  integer :: i,j ! loop variables
  integer dimension(1,nrecords:1,nvars) ! stores the data into the array vars
  open(101,file='all_possible_subsets.dat',status=old) ! open the hw datafile
  for i=1,nrecords,1 ! loop through all of the records in the file
  read(101,*)(vars(i,j) j=1,nvars,1) ! read each line
  end for
end program read_my_datafile
```

Center for the Study of Biological Complexity, Virginia Commonwealth University, Richmond, VA 23284-2030 US
E-mail address: tmwitten@vcu.edu