CS 3423.02

Assignment5: Implementing Word Count using multiple processes

Due: 12:05am of 11/23/2014

1 Overall Description:

This is a group project, with two members each group. You should find another member inside the class. Of course, you can do it by yourself if you are confident.

The basic idea of this project is to utilize the multiple processes to speed up the program that you just wrote in assignment4. The total target is similar to assignment4 and you can use one of group member's assignment4 implementation. **NOTE: it is not allowed to utilize the code of others inside the class.**
As assignment4, the target of this project is to find out the occurrences of each word (A Word Count Example) **inside a large file**, so that you don't need to handle the directories. Depending on whether you are using your designed doubly link list or not, you may turnin different files.

If you are using your designed doubly link list, then you should turnin the following files.

`wordcount.c, dllist.c and dlist.h.`

If you are using the STL library's link list, you will only turnin the `wordcount.cpp` file. Please refer to the following description about how to utilize the STL's link list, `http://www.cplusplus.com/reference/list/list/`.
NOTE: you will not get additional points for this project whether you are choosing your owned linked list or not since that is the target of last assignment. You can not add some additional files since I will compile your programs using my Makefile.

This program can accept four input parameters. If the number is less than or greater than 4 parameters, you program should handle that and print errors.

```
./wordcount INPUTFILE PROCNUMBER PRINTNUMBER OUTPUT
```

- The INPUTFILE parameter is a file, thus you will count the occurrence of words on this file.
- PROCNUMBER is to specify how many child processes that you will create to process the data. If the PROCNUMBER is 0, this is a valid and you don't create child processes.
- The PRINTNUMBER parameter is used to tell the program to output the NUMBER of words with most occurrences.
- We will redirect all words related information into a OUTPUT file, but only display the NUMBER of words on the screen. But those words has to be sorted at first before storing into the OUTPUT file.
- Each line in the OUTPUT file will only hold information of one word: the occurrences of this word, the word itself. These two fields are divided using the Table space, printing using the "/t". Those information is similar to the last two assignments.

NOTE: your program should be able to analyze the input parameters. You can't expect that I will input something in the middle.

1.1 Description of words

Checking the words is similar to what we did before.

- There is no difference between uppercase and lowercase. For example, "We", "we" and "WE" are considered the same and should be counted together. Thus, in end of output, we only output those number of occurrences of lowercase. If we have ("we", "we", "WE"), we will output:

  ```
  3 we
  ```

- We focus on those words made out of the alphabet, or connecting with hyphens. For example, "we're" will only count "we" here while "'re" will be discarded. However, "accident-prone" will be counted as a word. To be consistent with previous project, hyphen in anywhere will be considered as a valid word, such as "-good-word" or "goodword–".

- We can simply discard the following cases.
  - Anything that are not consisted of letters and hyphens. For example, words using with an apostrophe (') or even period. For example, "We all in this class.", there are only four words inside: "we all in this". "class." will be discarded.
  - Words made with numbers. For example, 1234 and 1234a will be discarded.
  - HTML tags

2 Implementation Tips

The following tips are just my suggestion. Feel free to use other designs and I only evaluate the `wordcount` file by providing with different parameters.

1. You may use `stat` to check the size of a total file, with `st_size` field.
2. Then you can use `mmap` to map this large file to the memory and divide the workload by passing different starting address and stopping address to different processes.
3. You will create different processes using `fork` system call. Normally, parent process (or the main process) is waiting there until all children processes exit.
4. Since results is located at different processes, which can't be seen by other processes, you can be creative in design an algorithm to sort different words in the end. You are free to use any way your want.

3 General Rule

- All programs should be able to handle those errors in the first place. Failing to do this will loose some points. For example, if an input has a problem, your program should be able to point out the errors.

  ```
  printf("ERROR: YOUR_SPECIFIC_DESCRIPTION\n");
  exit(1);
  ```

  Since we are going to check your error report using script, then you should use the specific format for this error report. The error report will always start as

  ```
  ERROR:
  ```

- Please pay attention to readablity of a program. You may refer to some coding style listed at `http://www.maultech.com/chrislott/resources/cstyle/indhill-annot.html`.
- You can change your code in assignmente to suit for this assignment.

3.1 Suggested Tests:

Make sure that you can verify your scripts using the following test cases:

1. The number of arguments is not correct. We only accept four arguments.
2. INPUT is the same as OUTPUT.
3. OUTPUT is already existing.
4. INPUT is a file.
5. File name can include space in the middle.

4 Submission Requirements

In this assignment, you will have to turnin two parts.

1. If you are using your designed linklist, you will turnin wordcount.c, dllist.c, dllist.h and Makefile. wordcount.c is the main program and you should not put `main` function into your dllist.c. If you are using STL link list, you only need to turn in wordcount.cpp and Makefile. (70%)
2. WCDesignForMultiprocess.pdf: more about this file are described below. Please use pdf format this time since you are required to put the performance figure inside. (30%)

For WCDesignForMultiprocess.pdf, you will have to include the following things.

1. **Name of collaborators**: Please include the ABCid and Name of both collaborators under the title.

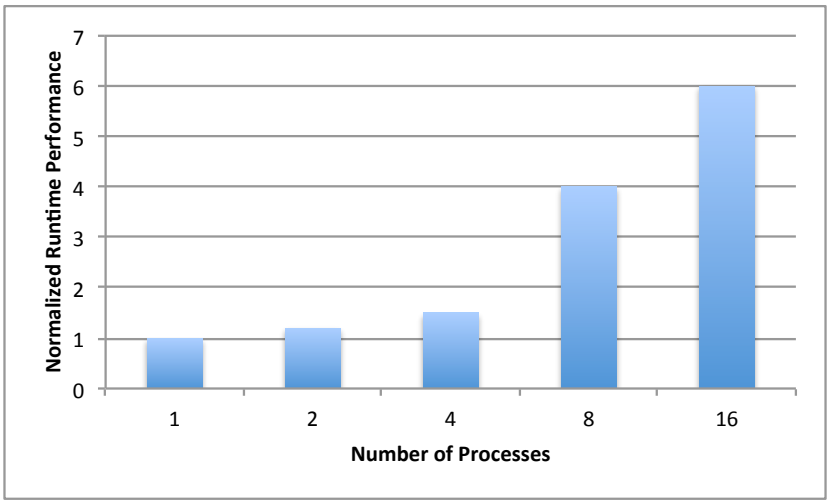

Figure 1: Performance of wordcount on different number of processes.

2. **Design Description:** This part describes how you implement your program, such as the steps to resolve this problem. You **should** describe the steps to finish this assignment. For example, how you can make different processes access different parts of the same file. How you define the relationship between the parent and children processes? How to know whether a child process has been exited or not? How to sort those words inside different processes? I should not READ your source code to know how you will finish this project. Please talk in a reasonably detailed, but not putting your code inside.

3. **Performance Comparison:** This part will get the performance of your program on a middle and large size input, which is provided at `http://www.cs.utsa.edu/~tongpingliu/teaching/cs3423/largeinput`. You will have to insert some figure as Figure 1.

   This time, you can evaluate the program by feeding different PROCNUMBER to it. In the end, you should provide a figure with different number of processes, at least including 1 (PROCNUMBER is 0), 2, 4, 8, 16 processes in the figure. You will normalize the performance to 1 process. Here is an example on this, if the runtime of 1 process is 1 second and the runtime of 2 processes is 0.5 second, then the normalized performance of 2 processes is 2 or 200%.

   In the meanwhile, **you should explain the reason of this performance data**. Horizontally, you will have to compare with middle input and large input. Vertically, you will explain the difference with different number of processes. You can get the CPU information by checking the following file:

   `/proc/cpuinfo`

4. **Advantage of Using Multiprocesses:** In which condition, do you think that multiple processes can provide you the best performance? You can be open minded here.

I expecting that the writeup is about or more than two pages.