INDIAN INSTITUTE OF TECHNOLOGY, KHARAGPUR
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
COMPUTING LAB I, AUTUMN 2016

Assignment 1: Understanding `Fork` System Call

July 21, 2016

1 PROBLEM STATEMENT

Objective of this assignment is to learn and explore the usage of `fork` system call. `fork` is used to create a child process and execute functions under that. For details of the `fork` functionalities, you may check the corresponding Linux man page.
For this assignment you will be given two input files containing a set of words, where there can be repeated entries of a single word. However, the words may not be in sorted order, and the repeated words may not come together. Your task is to find the set of unique words in each file along with the frequency of each unique word in each file. You also require to calculate the similarity between these two files in terms of Jaccard co-efficient.
You'll do the above tasks in two modes - **uni-process mode** where a single process will execute all the tasks sequentially, and **multi-process mode** where the text files would be divided in multiple chunks, and then one process will combine the results from others. Your task is to observe the differences in execution time for the uni-process mode and the multi-process mode.

1.1 INPUTS

The input to the system is as follows.

1. File-1
2. File-2
3. an integer k *(number of chunks each file will be divided)*

The two files contains a set of words, where the words may be repeated. The format for the input files are as follows. The first line of each file will contain an integer n (count of words) followed by n number of words, each at a single line.

1.2 Outputs

1. A set of unique words in file 1 , with their respective frequency
2. A set of unique words in file 2, with their respective frequency
3. Jaccard co-efficient for similarity measure

1.3 Example

File-1*:* {
7
flag
apple
institute
structure
analysis
airline
institute
}
File-2*:* {
8
nation
cricket
analysis
structure
digital
algorithm
flag
structure
}

OUTPUT: Unique words of File-1: {
flag(1)
apple(1)
structure(1)
analysis(1)
airline(1)
institute(2)
}

Unique words of File-2: {
nation(1)
cricket(1)
analysis(1)
structure(2)
digital(1)
algorithm(1)
flag(1)
}
similarity: $xx.xx$%

2 Details

As mentioned, the system will run either in uni-process mode (when $k = 1$) or in multi-process mode (when k>1).

Uni-process Mode: In uni-process mode when $k = 1$, the tasks are as follows.

T1 Find out the unique words from each file.

T2 Compute the frequency of those unique words from each file.

T3 Compute the similarity between the two files in terms of Jaccard Coefficient. Let A be the set of unique words in File-1 and B be the set of unique words in File-2. Then Jaccard Co-efficient ($J(A,B)$) is given as,

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Multi-process Mode: In multi-process mode when $k > 1$, you need to execute following steps.

S1 Divide each file into k chunks where $k > 1$. Let there be n number of words in a file. For dividing the files into k chunks, you take $\lceil \frac{n}{k} \rceil$ number of words in the first $k-1$ chunks, and the remaining $n - \lceil \frac{n}{k} \rceil$ number of words in the k^{th} chunk. For example, if there are 23 words in a file and $k = 4$, the you put 6 words in the first three chunks, and 5 words in the fourth chunk.

S2 For every file, generate k number of child processes using the `fork` system call, and provide one chunk to every child process. The child process will execute tasks T1 and T2, as mentioned before, only for those chunks and return the result to the parent. Note, as you have two input files, so you need to have $2k$ number of child processes.

S3 The parent process will then combine the results obtained from all children and find out the unique words and their occurrence frequencies in each file. Note, although you are counting the unique words for individual chunks, there can be scenarios where a word gets repeated under multiple chunks. Therefore, you need to develop a mechanism for the joining process.

S4 Finally compute the Jaccard co-efficient at the parent process for the two files using the unique words in each file.

Finally you need to compute the time for execution t_k for different values of k. Increase the values of k, execute your code and note down the time for execution. Plot a graph showing the value of t_k for $k = 1...10$.

3 DELIVERABLES

You need to submit the following files:

1. assign1_<Roll_No>.c
2. assign1_observation_<Roll_No>.txt
3. assign1_graph1_<Roll_No>.png – the graph showing evolution of t_k for different k.
4. the Makefile

You need to compress above files as *assign*1_*<Roll_No>.tar.gz* and submit this single compressed file in moodle by the deadline.
Deadline: 28 July 2016, 2:00 pm IST