Christopher Bradshaw
CSCI 4061
Assignment 2 Commentary

Files

This program consists of two files: parallel_convert.c and parallel_convert.h. The header file defines various constants, macros and function prototypes. The source file defines a main function and implements these function prototypes. A makefile is provided for compilation.

Program Execution Summary

Execution can be broken up into two stages: initialization/validation and the "main" conversion/HTML generation component.

1.) Initialization/Validation

At the beginning of the program, the process ID (PID) is stored. This is the main parent process. Following this, command line arguments are checked. There must be exactly 4 (executable name, convert count, output directory, input directory). A usage statement is printed if this is not satisfied. Following this, log files are initialized and the input/output directories are checked. The specified input directory MUST exist and be readable. The output directory will be created if it does not already exist. Following this, the input directory is analyzed. During this process, the parent process logs "junk" files in the specified output file and builds an array of valid images to convert. This array of valid files is used later by child processes when constructing the HTML files. This concludes the initialization of the program.

2.) File Conversion/HTML Generation

Following the initialization of the program, a while loop is entered-- the program remains in this loop until all input files have been processed. At the beginning of each loop iteration, a specified number of child processes are spawned. There are two ways in which the loop proceeds at this point: one way for a child process and the other for the parent process. The parent process simply waits for all children processes to finish. The children processes do the "heavy lifting" of the program-- that is, they handle file conversion. The child process selects a file type to handle based on its PID. If an unmanaged file of this type is not found, it simply exits. A process selects a file to handle by traversing the input directory and searching for the first file which satisfies the following conditions: 1.) This file has the type that this process is meant to handle ; 2.) This file has not been converted by another process (by checking the output directory). Once a file matching these criteria has been found, a HTML file is generated for it. The array of valid files (made during the initialization section) is searched for this file. The generated HTML file for this image links to the next file in this list (or the first in the case that this file is the last in the array). Once the HTML file has been generated, the conversion is facilitated by using *execlp(), which* takes over program execution and exits once it has completed.