Parallel-Primes

Jordon Biondo

September 26, 2013

1 Design Document

1.1 Data Structures

The program does not make use of any special data structures.

1.2 Flow Control

The program makes use of macros specifically made for flow control during forks as seen in the following example. This is key to the programs design.

```
// fork and set *child_pid to the new pid, then send each process to the specified targ
FORK_TO(&child_pid, on_parent, on_child, on_error);

// During normal execution these blocks will be ignored,
// they are only accessible through gotos.
target(on_parent) {
  printf("parent process here");
  shoot_to(end);
}

target(on_child) {
  printf("child process here");
  shoot_to(end);
}

target(on_error) {
  printf("forking error!");
  shoot_to(end);
```

```
}

target(end) {
  printf("done!");
}
```

1.3 Algorithm

If a new process's memory is thought of like a new stack frame, the program uses a somewhat recursive algorithm. Each process, including the parent process does the following:

- If the process is the initial root process, its prime value is set to 2
- Child PID and Pipe information is cleared in the process
- A pipe is created for the coming fork.
- The process forks, the parent and child processes are sent to different blocks of code.
- The child sets its input to the pipe's output, the parent set's it's output to the pipe's input.
- A parent process will read in numbers from a function repeatadly. This function is called as a function pointer, the root parent process has a pointer that returns an ever increasing static number. All subsequent parent processes have a pointer to a function that reads from their stdin file descriptor (the pipe). If the number read is not divisible by the parent's prime it is sent to the child through the pipe.
- A child process will stop and read one number from the pipe, this is that childs prime. If in receiving this prime the child has not reached the stop conditions, the child will goto the beginning of the program again and will repeat what the initial process did.
- If stop conditions are met, the child process which will cause it's parent to stop writing and exit, this goes all the way up the chain to the initial root process.