UNIVERSIDADE FEDERAL DE OURO PRETO

DEPARTAMENTO DE COMPUTAÇÃO

CIÊNCIA DA COMPUTAÇÃO

SISTEMAS OPERACIONAIS – BCC 264

PROFESSOR: CARLOS FREDERICO MARCELO DA CUNHA CAVALCANTI

ALUNA: ANANDA MENDES SOUZA

TRABALHO PRÁTICO II

1. As diferenças que você encontrou entre threads e "fork".

Quando vários processos rodam concomitantemente no mesmo processador, o kernel do SO pode dividi-los em unidades de processamento que podem ser agendadas segundo uma multiplicação por fração de tempo (time-slice multiplexing). Estas unidades são chamadas de threads de um processo. Processos podem rodar várias threads concorrentemente (multithread processo) que otimiza a utilização da memória. Threads de um processo podem compartilhar recursos e memória, o que dois processos não podem fazer, por terem address space diferentes.

A maneira como o SO lida com os processos depende de como o processo foi programado e como ele realiza as chamadas de sistema (system call). Quando dois processos iguais correm em um mesmo sistema Unix, o kernel faz o fork() do processo e cria um novo processo (processo filho) com um novo address space. Além disso, o fork precisa do IPC (InterProcess Comunication) para trocar informações entre o processo pai e o processo filho, depois do fork. Alternativamente, o SO pode usar o recurso de múltiplos threads de um mesmo processo. Isto é mais vantajoso porque a comunicação e compartilhamento de recursos entre as threads é mais fácil (por estarem no mesmo address space).

Existe, porém, um problema decorrente da chamada de um fork() em um processo multithread. Por exemplo, em um processamento multitarefa um fork() registra no processo pai o PID do processo filho. Mas, quando um thread em uma tarefa multithread executa um fork() qual delas conteria o processo filho? Porém é uma situação evitada sempre que

possível, existem situações onde o fork do processo multithread resulta no processo filho (também multithread) e o término do processo filho precisa retornar informação para o processo pai via IPC.

Portanto, pode ser que o processo filho contenha um thread correspondente ao thread do processo pai.

2. Como usar o C-Compiler.

É um compilador e depurador online de C/C++. Para utilizá-lo basta copiar o código e colar no site e depois clicar em run. Uma característica importante é poder usar a parte de baixo como prompt de comando, o que facilita bastante para digitar as operações do código.