GSI018 – SISTEMAS OPERACIONAIS

Operating Systems – William Stallings – 7th Edition
Chapter 04 – Processes and Threads

Pedro Henrique Silva Santana – 12011BSI218 – pedro.santana@ufu.br
Victor Hugo Martins Alves – 12011BSI217 – victor.alves1@ufu.br

PROBLEMS

4.1 It was pointed out that two advantages of using multiple threads within a process are that (1) less work is involvedin creating a new thread within an existing process than in creating a new process, and (2) communication among threads within the same process is simplified. Is it also the case that a mode switch between two threads within the same process involves less work than a mode switch between two threads in different processes?

Verdadeiro. Threads com mesmo processo possuem maior facilidade de comunicarem entre si pois não é necessário invocar o kernel.

4.2 In the discussion of ULTs versus KLTs, it was pointed out that a disadvantage of ULTs is that when a ULT executes a system call, not only is that thread blocked, but also all of the threads within the process are blocked. Why is that so?

As threads em User-Level Threads não conseguem ser escalonadas assim como no Kernel-Level Threads.

4.4 Consider an environment in which there is a one-to-one mapping between user-level threads and kernel-level threads that allows one or more threads within a process to issue blocking system calls while other threads continue to run. Explain why this model can make multithreaded programs run faster than their single-threaded counterparts on a uniprocessor computer.

De acodo com o sistema apresentado, o sistema de bloking não poderia bloquear por completo a execução pois o kernel thread é presente em toda user thread. Então caso uma kernel thread fosse bloqueada, as demais continuariam a rodar.

Já o sistema de uniprocessor, seria necessário esperar pela conclusão do I/O operation na maioria das vezes. Assim o sistema de multithreaded teria vantagem no tempo de execução.

4.5 If a process exits and there are still threads of that process running, will they continue to run?

Não, pois todas as threads que compartilham do mesmo espaço de endereço são encerradas no mesmo tempo.

4.7 Many current language specifications, such as for C and C++, are inadequate for multithreaded programs. This can have an impact on compilers and the correctness of code, as this problem illustrates. Consider the following declarations and function definition:

```
int global_positives = 0;
  typedef struct list {
  struct list *next;
  double val;
} * list;

void count_positives(list l) {
  list p;
  for(p = l; p; p = p -> next )
    if( p -> val > 0.0 )
      ++global_positives;
}
```

Now consider the case in which thread A performs “count_positives(<list containing only negative values>)” while thread B performs “++global_positives”.
a. What does the function do?

A função para a thread A irá percorrer por toda lista para verificar se possui algum valor positivo. Já a thread B não será acionada pois a lista não contem positivos a serem incrementados pela variável “global_positives”.

b. The C language only addresses single-threaded execution. Does the use of two parallel threads create anyproblems or potential problems?

Não, pois mesmo em um sistema de único processador, o SO pausa a thread A para rodar a thread B quando necessário, porem mesmo assim não seria necessário visto que a thread a possui apenas valores negativos, assim não invocando a thread B.