Paradigmas de Computação Paralela

17-Nov-2015

Memory Consistency Models

These exercises aim to promote a better understanding of OpenMP memory consistency model.

1) Execute the following OpenMP program and explain the program output.

```
#include <stdio.h>
#include <omp.h>
int main(){
int x;
    x = 2;
    #pragma omp parallel num_threads(2) shared(x)
    {
        if (omp_get_thread_num() == 0) {
        x = 5;
        } else {
            /* Print 1: the following read of x has a race */
            printf("1: Thread# %d: x = %d\n", omp_get_thread_num(),x );
        }
        #pragma omp barrier
        if (omp_get_thread_num() == 0) {
            /* Print 2 */
            printf("2: Thread# %d: x = %d\n", omp_get_thread_num(),x );
        } else {
            /* Print 3 */
            printf("3: Thread# %d: x = %d\n", omp_get_thread_num(),x );
        }
    }
    return 0;
}
```

2) Complete the following OpenMP programs with “flush” directives to ensure the correct running behavior.

```
#include <omp.h>
#include <stdio.h>
int main()
{
    int data;
    int flag=0;
    #pragma omp parallel
    {
        if (omp_get_thread_num()==0)  {
            /* Write to the data buffer that will be  read by thread */
            data = 42;
            /* Set flag to release thread 1 */
            flag = 1;
        }
        else if(omp_get_thread_num()==1)
        {
            /* Loop until we see the update to the flag */
            while (flag < 1)
            {
            }
            printf("flag=%d data=%d\n", flag, data);
        }
    }
}
```

```
#include <omp.h>
#include <stdio.h>
int main()
{
    int flag=0;
    #pragma omp parallel num_threads(3)
    {
        if(omp_get_thread_num()==0)
        {
            /* Set flag to release thread 1 */
            #pragma omp atomic
            flag++;
        }
        else if(omp_get_thread_num()==1)
        {
            /* Loop until we see that flag reaches 1*/
            while(flag < 1)
            {
            }
            printf("Thread 1 awoken\n");
            /* Set flag to release thread 2 */
            #pragma omp atomic
            flag++;
        }
        else if(omp_get_thread_num()==2)
        {
            /* Loop until we see that flag reaches 2 */
            while(flag < 2)
            {
            }
            printf("Thread 2 awoken\n");
        }
    }
}
```