NAG Library Function Document

nag_omp_set_nested (x06agc)

1 Purpose
nag_omp_set_nested (x06agc) enables or disables nested OpenMP parallelism.

2 Specification
#include <nag.h>
#include <nagx06.h>
void nag_omp_set_nested (Integer nesting)

3 Description
nag_omp_set_nested (x06agc), for multi-threaded implementations, enables or disables the nesting of
OpenMP parallel regions by setting an OpenMP Internal Control Variable (ICV) and any vendor library
specific options where that is possible. See the Users’ Note for your implementation for details of the
scope of nag_omp_set_nested (x06agc).
Nesting is disabled by default in OpenMP.
In serial implementations of the NAG C Library nag_omp_set_nested (x06agc) has no effect. See the
Chapter x06 for a discussion of the behaviour of these functions when called in serial.

4 References
OpenMP Specifications http://openmp.org/wp/OpenMP-Specifications
Programming The MIT Press

5 Arguments
1: nesting – Integer
   Input
   On entry: if nesting = 0, nesting of OpenMP parallel regions is disabled, otherwise it is enabled.

6 Error Indicators and Warnings
None.

7 Accuracy
Not applicable.

8 Parallelism and Performance
Not applicable.

9 Further Comments
None.
10 Example

In this example we presume a multi-threaded implementation of the NAG Library. We call nag_omp_set_nested (x06agc) to enable the nesting of OpenMP parallel regions by setting an ICV. Inside a first, or ‘outer’, parallel region we call nag_omp_get_nested (x06ahc) to display the status of nesting, and expect to see it return the value 1.

Each thread then spawns a second ‘inner’ parallel region. Within this region we have each thread increment the value total. As there are 5 threads in the outer region and 3 threads in each inner region we expect the final value of this variable to be 15.

If you use a serial implementation of the NAG library, regardless of whether the code is compiled with OpenMP or not, nag_omp_set_nested (x06agc) will have no effect and nag_omp_get_nested (x06ahc) will always return 0. The total number of threads will be 1. The appropriate results file will be included with the distribution material for your implementation.

10.1 Program Text

/* nag_omp_set_nested (x06agc) Example Program. */
/* Copyright 2014 Numerical Algorithms Group. */
/* Mark 25, 2014. */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagx06.h>

int main(void)
{

    /* Scalars */
    Integer exit_status = 0;
    Integer nesting, nesting_set, num_inner, num_outer, total;

    /* Nag Types */
    NagError fail;

    INIT_FAIL(fail);

    /* Output preamble */
    printf("nag_omp_set_nested (x06agc)\n");
    printf(" Example Program Results\n\n");

    nesting_set = 1;
    /* nag_omp_set_nested (x06agc).
     * Enable the nesting of OpenMP parallel regions
     */
    nag_omp_set_nested(nesting_set);

    num_inner = 3;
    num_outer = 5;

    /*
     * num_omp_set_num_threads (x06aac).
     * Set the number of threads for an outer parallel region
     */
    nag_omp_set_num_threads(num_outer, &fail);

    if (fail.code != NE_NOERROR) {
        printf("Error from nag_omp_set_num_threads (x06aac).\n", fail.message);
        fflush(stdout);
        exit_status = 1;
        goto END;
    }
/* Spawn an OpenMP parallel region and have the master thread check whether
 * nesting of parallel regions has been enabled
 */

int total = 0;

#pragma omp parallel shared(num_inner,total) private(fail,nesting) \
    reduction(+:exit_status) default(none)
{
    INIT_FAIL(fail);
    
    /*
     * nag_omp_get_nested (x06ahc).
     * Test whether nesting of parallel regions has been enabled
     */
    nesting = nag_omp_get_nested();

    #pragma omp master
    {
        printf("\n%s %11" NAG_IFMT " \n\n", "Nesting enabled: ", nesting);
    }
    
    /*
     * Set the number of threads for an inner parallel region
     */
    nag_omp_set_num_threads(num_inner, &fail);

    if (fail.code != NE_NOERROR)
    {
        printf("Error from nag_omp_set_num_threads (x06aac).\n%s\n", fail.message);
        exit_status++;
    }
    else
    {
        /*
         * Spawn a nested parallel region and add up the total number of threads
         * in all teams
         */
        #pragma omp parallel shared(total) default(none)
        {
            #pragma omp atomic
            total++;
        }
    }
}

fflush(stdout);

if (exit_status == 0)
{
    printf("\n%s %11" NAG_IFMT " \n\n", "Total number of threads requested ", num_outer * num_inner);
    printf("\n%s %11" NAG_IFMT " \n\n", "Total number of threads from sum: ", total);
    fflush(stdout);
}

END:
return exit_status;
10.2 Program Data
None.

10.3 Program Results
nag_omp_set_nested (x06agc) Example Program Results

Nesting enabled: 0

Total number of threads requested 15

Total number of threads from sum: 1