Banded Matrix-vector product

1 Banded Matrix-vector product

This exercise is about parallelizing the product of a banded matrix times a dense vector. A banded matrix A is such that, for a *bandwidth* b, all coefficients $A_{i,j}$ with $|i-j| > b$ are equal to zero. This means that on row i only the coefficients

$$a_{i,i-b}, ..., a_{i,i-1}, a_{i,i}, a_{i,i+1}, ..., a_{i,i+b}$$

are nonzero. The example below shows a matrix of size $n = 6$ with $b = 2$.

$$\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & 0 & 0 & 0 \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} & 0 & 0 \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} & a_{3,5} & 0 \\ 0 & a_{4,2} & a_{4,3} & a_{4,4} & a_{4,5} & a_{4,6} \\ 0 & 0 & a_{5,3} & a_{5,4} & a_{5,5} & a_{5,6} \\ 0 & 0 & 0 & a_{6,4} & a_{6,5} & a_{6,6} \end{bmatrix}$$

If the matrix is very large and the bandwidth relatively small, i.e. $b \ll n$, it may be worth to store the matrix in compact storage, where most of the zero coefficients are ignored. One option is to store the matrix in an array with $2 * b + 1$ rows and n columns such that each row contains a diagonal of the matrix and each column contains on column. The matrix above can be stored like this

$$\begin{bmatrix} 0 & 0 & a_{1,3} & a_{2,4} & a_{3,5} & a_{4,6} \\ 0 & a_{1,2} & a_{2,3} & a_{3,4} & a_{4,5} & a_{5,6} \\ a_{1,1} & a_{2,2} & a_{3,3} & a_{4,4} & a_{5,5} & a_{6,6} \\ a_{2,1} & a_{3,2} & a_{4,3} & a_{5,4} & a_{6,5} & 0 \\ a_{3,1} & a_{4,2} & a_{5,3} & a_{6,4} & 0 & 0 \end{bmatrix}$$

Note that this storage requires some artificial zero coefficients to make all the column of the same height.

In this exercise we are interested in computing the product $y = A * x$ where the matrix A is stored in compact form. This can be done either by traversing the coefficients of the compact form column by column (which corresponds to traversing the coefficients of A column by column) or row by row (which corresponds to traversing the coefficients of A one diagonal at a time). We will parallelize these two variants of the product.

2 Package content

In the `spmv` directory you will find the following files:

- `main.c`: this file contains the main program which first calls the `init_data` routine which generates a random sparse matrix of size n and bandwidth b and initializes with random values the vectors `x` and `y`. The program first performs the matrix vector product using the standard "full matrix" format (this is only provided as a reference) and then using the two variants described above which are based on the compact storage. For each of these two versions, the main program checks that the result is correct. **Only this file has to be modified for this exercise**.

- `aux.c, aux.h`: these two files contain auxiliary routines and **must not be modified**.

The code can be compiled with the `make` command: just type `make` inside the `band_matrix` directory; this will generate a `main` program that can be run like this:

```
$ ./main n b
```

where `n` is the number of rows and columns in the matrix and `b` is the bandwidth.

3 Assignment

- Parallelize the `matmul_compact_row` and `matmul_compact_diag` routines described above.

  For both versions developed above, make sure that the result is correct.

- Report the execution times for the implemented parallel versions with 1, 2 and 4 threads and compare them. What speedup could you achieve? Which version is faster and why? analyze and comment on your results for different values of n and b. Report your answer in the `responses.txt` file.