Parallel assembly of sparse matrices using OpenMP

5x faster than MATLAB
On 6 cores our program runs more than 5 times faster than Matlab sparse function and twice as fast as fsparse.

Our goal of reaching a speed of 5 times greater than Matlab is thus accomplished.

The peak performance
Our program peaks at around 16 cores but after 6 cores there is no big win. 6 cores is not unusual inside personal computers, which means this performance boost is available to all.

All the tests are made on unsorted input arrays.

Importance of sparse matrices
Assembling sparse matrices is very common in finite element methods. Each node in the mesh corresponds to a matrix element and thus a sparse matrix is suitable.

A new sparse matrix is generated every time the mesh changes. This occurs for every time step of a typical simulation. Speeding the process up is thus desirable.

More is possible
Our program can be improved further by optimizing the OpenMP clauses. This can probably improve performance slightly. Changing the algorithm of fsparse however might improve performance drastically.

OpenMP: A great tool
A parallel language made to specify parallel sections in the code while the rest runs sequentially, as in the figure to the right, and the example to the left.

By applying OpenMP to our sequential code we could boost its performance significantly. Thanks to OpenMP being very portable and friendly, every user can now achieve the same boost in performance as us easily on their own PC.