TOWARDS HPC-BASED SHALLOW FLOW SIMULATIONS FOR VARIOUS PROBLEMS

Bobby Minola Ginting
Chair for Computation in Engineering, Technical University of Munich, Germany
bobbyminola.ginting@tum.de

This contribution presents successful results as well as future implementations of the author's works for various simulations of shallow flows with hybrid OpenMP-MPI. To enable such simulations, the shallow water equations (SWEs) are solved. In the past decade, the SWEs models have been proven to be successful to simulate the complex environmental systems on large domains, e.g. urban flood and tsunami simulations. For large scale applications, the role of high-performance computing (HPC) is highly important to significantly reduce computational time enabling immediate measures. Interestingly, the SWEs can also be extended in such a way, e.g. by adding a two-equation turbulence model, so that the flow properties such as vortex shedding may be captured accurately with high-resolution meshes. To this end, HPC is obviously beneficial to reduce computational time. We perform our simulations by means of an in-house code NUFSAW2D (*Numerical Simulation of Free Surface Shallow Water 2D*) employing an edge-based cell-centered finite volume (CCFV) model. This code has been tested for various benchmark cases, which range from small scale applications, e.g. turbulent flows around a conical island to capture the vortex shedding with wet-dry phenomena, turbulent dam-break flows with scalable wall functions, etc. – to the large ones, e.g. real urban flood and tsunami simulations in some countries. Due to their flexibility, unstructured meshes including combinations of triangular and quadrilateral cells are used in our works, thus enabling accurate simulations on very complex domains. Yet, decomposing such domains turns out to be a quite challenging task, particularly for MPI, as the domains must be decomposed and allocated to different processors. Although some tools are readily available (e.g. METIS library) or some advanced techniques such as space-filling curves (SFCs) can be employed to decompose the domains, these ways might lead to thousands of different scenarios of obtaining the most efficient results. We observe that this matter is a trade-off between the communication patterns among nodes and the memory access patterns inside each node in our implementations. Therefore, relying only on one tool/strategy for decomposing the domains is not an option. To this purpose, we have designed a simple and flexible data structure in our code to support any type of reordering tool and to account for various future applications. In this way, users may have to, before performing a simulation, consider the domain properties (the total numbers of edges and cells) only, without having to think how the domain will be decomposed. At this stage, NUFSAW2D will always provide contiguous 1D-array configurations easing edge/cell allocation among nodes, while both communication and memory access patterns still remain unknown. In the next stage, one can then employ any suitable tool/strategy to obtain such patterns. Another essential part in our current work lies on optimizing the load balance among threads and nodes, with emphasis on wet-dry problems. In most real shallow flow applications, wet-dry problems exist – and in the framework of HPC, such problems cause load imbalance among processors as the wet cells require more computational efforts than the dry ones. Even if the most ideal decomposition for a static or an adaptively-refined domain can be obtained, the performance may significantly decrease due to the existence of wet-dry problems, which cannot exactly be predicted during runtime. To anticipate this problem, we have developed a novel weighted-dynamic load balancing (WDLB) strategy that distinguishes the wet cells from the dry ones based on their complexity level. The core idea of this strategy is to detect at a certain time level the existence of wet cells (thus a load unit can be assigned to such cells) firstly in the thread level – and later all load units among threads are collected to each node level. Afterwards, all load units among nodes are summed up, yielding a total amount of load units for the entire domain. Now, the reverse procedures must be carried out, where the total load units are distributed back according to the total number of nodes; once the new load units have been assigned to each node, they are now distributed to each thread. With this concept, even if using a static domain, the computation loads are dynamically distributed during runtime. This strategy is quite simple and can readily be applied to any CCFV scheme. Some future implementations of our work are also presented here in the scope of HPC, such as non-hydrostatic turbulent wet-dry simulations for tsunami events in various scales, which require the WDLB strategy and the linear system computations to be performed efficiently in parallel – and adaptive mesh refinement (AMR) simulations with wet-dry problems, for which a new load balancing strategy may be required, for example the extension of the WDLB technique. This work would be useful for practical engineering purposes in the scope of HPC-based environmental modeling.

Keywords: HPC, hybrid OpenMP-MPI, shallow flow, weighted-dynamic load balancing