Review of: Tree based mesh-refinement GPU accelerated tsunami simulator for real time operation

Marlon Arce Acuña, Takayuki Aoki

Overview

In the current high-performance computing (HPC) context, where accelerated computing is revolutionizing not only the computation capacity but also the computation time required to perform the numerical simulation of large problems, it's very important the development of new numerical models able to change the game rules in the context of natural hazard early warning. In this framework these authors develop a new tsunami model (TRITON-G) coded in GPU architecture. The model is based on Spherical non-linear shallow water equations (SSWE) in the far-field area while the near-field and inundation areas the cartesian coordinate version of SWE is used. The developed numerical model in the far-field is based on the method of characteristics with a cubic polynomial approximation on the grid. In the near-field area, when inundation is calculated, they use a finite volume model. Authors use a quad-tree refinement method to build the grid for the computational domain. Later, it's explained in detail the GPU and multi-GPU implementation and finally they present some numerical experiments based on analytical solutions and a simulation of the Indian ocean 2004 tsunami. The article closes with final conclusions and a list of 67 references.

Overall Recommendation

My recommendation is: major revision.

Assessment and Further comments

In the introduction, authors take a tour of the different tsunami forecasting models by classifying them in two groups: depth-averaged non-hydrostatic and hydrostatic models. Although authors discuss briefly about some well-known in the tsunami context models, I don't understand the criteria followed for leaving out some important models, especially those models used by different Tsunami Warning Centers along the world: that is the case of COMCOT (http://223.4.213.26/archive/tsunami/cornell/comcot.htm), NEOWAVE (Non-hydrostatic Evolution of Ocean WAVE) or even in the GPU framework, Tsunami-HySEA (https://edanya.uma.es/hysea/index.php/2015-10-08-15-52-45/tsunami-hysea) or more recently the GPU version of NAMI DANCE (http://namidance.ce.metu.edu.tr/). Of course, it's not possible to include all the developed tsunami models, but as the development of TRITON-G was focused on its operational capabilities, it would be expectable to focus this analysis on models used in this context, moreover if some these models have been developed in the GPU framework (for example: the MOST model -included in the paper-, but that has been fully re-coded in GPU-, or Tsunami-HySEA that is being used by the Centro di Allerta di Tsunami (CAT) by the INGV (Italy) in the NEAMTWS context or the Joint Research Center of the European Union through the Global Disasters Alert and Coordination System (GDACS) (http://meetingorganizer.copernicus.org/EGU2015/EGU2015-13797-3.pdf)

Section 2

In section 2, authors describe the spherical non-linear SWE used in the tsunami-propagation and refer to Toro, 2010 for the cartesian coordinate version of the SWE used when inundation needs to be computed. This election will mean the use of different numerical methodology (described in section 3) depending on the area where computations are performed. So, regarding the numerical methods used in this paper: in the propagation-stage it's used the method of characteristics (MOC) in combination with a cubic-polynomial interpolation to find the interpolated values on the grid. Authors recall that the numerical method described is well-balanced but on other hand I'm missing two other important features that should be proven: conservation of mass and water height positivity. Is the total mass conserved? For instance, what about the mass conservation in the experiment described in p8 line 10? Regarding the water height positivity (h), is it guaranteed in the propagation stage even considering the cubic-polynomial interpolation process?

On other hand, what is the convergence order of the numerical models used on each area?

Section 3

As the coordinate system used combines spherical coordinates and cartesian coordinates, how is treated the boundary between the considered domains on each coordinate system?

Regarding the run-up calculation, in section 3.2 authors propose a technic based on considering (hu=hv=0) when h is less than a certain small fixed quantity. They confirm that the proposed implementation has been proven to be robust and stable under different benchmarks. Please, include a reference where this numerical technic is used.

Finally, in this section it's included a parabolic bowl problem as validation problem. That's a good synthetic test but I consider that it's not enough to define a numerical model as validated when such numerical model that is going to be used with real topo-bathymetries in real cases, moreover when, as it's remarked by authors, the model is going to use in the RIMES context. In order to validate this numerical scheme to be used in real cases I suggest, for instance, the use of the inundation benchmark experiments proposed by the National Tsunami Hazard Mitigation Program (NTHMP) in http://nws.weather.gov/nthmp/documents/nthmpWorkshopProcMerged.pdf or [Horrillo, J., Grilli, S.T., Nicolsky, D. et al. Pure Appl. Geophys. (2015) 172: 869. https://doi.org/10.1007/s00024-014-0891-y] or [Macías, J., Castro, M.J., Ortega, S. et al.

Pure Appl. Geophys. (2017) 174: 3147. https://doi.org/10.1007/s00024-017-1583-1] where problems with analytical solutions, laboratory experiments and real problems with field measurements are proposed. I would suggest studying the behavior of TRITON-G at least in the "mandatory" benchmark problems: BP1, BP4, BP6, BP7 and BP9.

Regarding the field measurement experiments, authors show in section 6.1.2 some inundation maps in different locations (Fig. 18 and 19) where the comparisons are made basically against other models. I think that it cannot be considered a numerical model as validated with field measurements by making comparisons basically with other models results. There are available many tsunami field measurements to validate the inundation process. In this sense you can consider cases where there are more available data than in the case studied in section 6.1.2. For instance: BP7, BP9 or many inundation scenarios related to the Tohoku, 2011 event where detailed data are available.

Regarding the generation process described in section 3.3, authors use the coseismic deformation proposed by Smylie and Manshiha, 1971. I'm surprised at this point because in [Okada, Y. (1985), Surface deformation due to shear and tensile faults in a half-space, Bulletin of the Seismological Society of America, 75(4), 1135-1154] this work is extended, and it's provided also an analytical way to get the "Okada model" coseismic deformation that can easily be extended to CUDA in the GPU context. I would like to know the reason for not considering Okada as model for the generation process.

Section 4

In section 4, authors use a tree-based mesh refinement similar to the AMR technic used by R.J. LeVeque in GeoClaw. In this case the refinement is not made automatically as in GeoClaw, but it's customized to be refined in coastal areas and focal areas. It's an interesting alternative to the use of nested meshes when more detailed information is required in certain coastal areas. In p14, line 9-10 it is discussed the number of blocks necessary under the considered resolution and according the 65 x 65 node-centered cells if the number of blocks is 230,000 you would have 971,750,000 node-centered cells. How many information is stored on each node-centered cell (in double precision) to represent over 100GB of memory space?

Section 5

Regarding section 5, I have some comments, but my main concern is that I think that this section is out of the scope of this journal and my recommendation would be to publish it in a journal more related to this field. Anyway, I will make some comments regarding this section. To my knowledge the overall GPU implementation has been solved in a very efficient way, particularly the implementation of the pipe asynchronous output that is crucial to deal with the GPU-CPU traditional bottleneck. In p23, line 8 it is showed that you use CFL 0.8, is not stable the implemented model for CLF nearer to 1? On other way, in p24, line 5 it is used **Δ**t = 1.6 for blocks with levels over 3. Is this consistent with the

CFL condition? Or, can you ensure the stability of the numerical scheme under this assumption?

In section 5.3.2 it is showed the runtime performance by comparing results obtained with two different GPU architectures: Tesla K80 and Tesla P100. It must be remarked that the configuration of the computation nodes that you are using is different for each architecture. While the Tesla P100 nodes have 4 GPU's per node, the Tesla K80 nodes have 2 GPU's per node. Unless your tests are using only two P100 per node, the network communications will reduce the performance between the K80 in front of P100. What kind of network is being used in these clusters? Another point is, are output data being stored into hard disks in these tests or are they related only to pure computation time?

Section 6

Finally, my main comments about section 6 have been included in my recommendations for section 3. Anyway, I have some specific comments about this section. It would be nice if the color scale used in Fig. 18 and 19 are the same for each subplot. With these graphics we can compare the inundation extensions, but it's difficult to compare the inundation height when different scales are used on each graphic. On other hand, and as I pointed before, this test is not enough to say that the model is validated, so I don't agree with the sentence of p33, lines 8-9.