The Shallow Water Equations and CUDA

Daniel Butnaru, Christoph Kowitz

December 5

Improvement of the MM multiplication

- coalesced memory access
- loop unrolling
- thread granularity

Performance Measures

- runtime
- occupancy
- warp serialize
- instruction throughput

What can still be done?

binary fan in for reduction

New Playground: Shallow Water Equations

- used for the simulation of layers of fluid
- applicable when horizontal scales are much larger than vertical scales
- derived from Navier-Stokes equations

Applications

- tsunami simulations
  - the ocean is not shallow, but wide
  - whole water column is influenced
- atmospheric simulations
  - for whole planets

Your Exercise

- use CUDA to efficiently calculate the SWE's
- framework will be given

Shallow Water Equations

$$\frac{\partial h}{\partial t} + \frac{\partial (v_x h)}{\partial x} + \frac{\partial (v_y h)}{\partial y} = 0$$

$$\frac{\partial (h v_x)}{\partial t} + \frac{\partial (h v_x v_x)}{\partial x} + \frac{\partial (h v_y v_x)}{\partial y} + \frac{1}{2} g \frac{\partial (h^2)}{\partial x} = -gh \frac{\partial b}{\partial x}, \qquad (1)$$

$$\frac{\partial (h v_y)}{\partial t} + \frac{\partial (h v_x v_y)}{\partial x} + \frac{\partial (h v_y v_y)}{\partial y} + \frac{1}{2} g \frac{\partial (h^2)}{\partial y} = -gh \frac{\partial b}{\partial y},$$

with

h	water height over ground
v_x, v_y	velocity in x and y direction
g	earths gravitation acceleration
b	bathymetry (height of the sea ground)
$hv_i = p, q$	linear momentum of the fluid

Spatial Discretetization

- cartesian equidistant grid
- all data a centered in the nodes (p, q, h)

Time Discretization

- forward euler
- variable timesteps

$$\frac{h_{ij}^{(n+1)} - h_{ij}^{(n)}}{\tau} + \left.\frac{\partial p}{\partial x}\right|_{ij}^{(n)} + \left.\frac{\partial q}{\partial y}\right|_{ij}^{(n)} = 0$$

$$\frac{p_{ij}^{(n+1)} - p_{ij}^{(n)}}{\tau} + \left.\frac{\partial (v_x p)}{\partial x}\right|_{ij}^{(n)} + \left.\frac{\partial (v_y p)}{\partial y}\right|_{ij}^{(n)} + \frac{1}{2} g \left.\frac{\partial (h^2)}{\partial x}\right|_{ij}^{(n)} = -gh \left.\frac{\partial b}{\partial x}\right|_{ij}$$

$$\frac{q_{ij}^{(n+1)} - q_{ij}^{(n)}}{\tau} + \left.\frac{\partial (v_x q)}{\partial x}\right|_{ij}^{(n)} + \left.\frac{\partial (v_y q)}{\partial y}\right|_{ij}^{(n)} + \frac{1}{2} g \left.\frac{\partial (h^2)}{\partial y}\right|_{ij}^{(n)} = -gh \left.\frac{\partial b}{\partial y}\right|_{ij}$$

Computing Spatial Derivatives

Finite Differences

Computing Spatial Derivatives

Finite Differences

- compute derivative from cell neighbours

$$\frac{\partial p}{\partial x} \approx \frac{p_{i+1} - p_{i-1}}{2\Delta x} \tag{2}$$

Computing Spatial Derivatives

Finite Differences

- compute derivative from cell neighbours

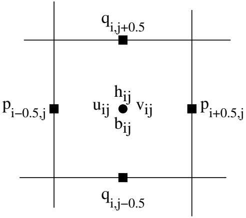
$$\frac{\partial p}{\partial x} \approx \frac{p_{i+1} - p_{i-1}}{2\Delta x} \qquad (2)$$

Finite Volume

- computation by evaluation of the fluxes over cell edge
- change of waterheight (h) can depends on the influx and outflux (hv)

$$\left.\frac{\partial p}{\partial x}\right|_{ij}^{(n)} \approx \frac{p|_{i+\frac{1}{2},j}^{(n)} - p|_{i-\frac{1}{2},j}^{(n)}}{\Delta x} \quad \text{and} \quad \left.\frac{\partial q}{\partial y}\right|_{ij}^{(n)} \approx \frac{q|_{i,j+\frac{1}{2}}^{(n)} - q|_{i,j-\frac{1}{2}}^{(n)}}{\Delta y},$$

Evaluation of the Flux on the Edge



Taking Averages on the edge

- average the momentum flux between to cells

$$p|^{(n)}_{i+\frac{1}{2},j} = \frac{1}{2}\left(p^{(n)}_{ij} + p^{(n)}_{i+1,j}\right) \quad \text{or} \quad q|^{(n)}_{i,j-\frac{1}{2}} = \frac{1}{2}\left(q^{(n)}_{i,j-1} + q^{(n)}_{ij}\right)$$

Lax-Friedrich Flux Computation

due to stability issues the computed flux has to be corrected to accurate terms by an averaging over the flux density.

$$p|_{i+\frac{1}{2},j}^{(n)} = \frac{1}{2}\left(p_{ij}^{(n)} + p_{i+1,j}^{(n)}\right) + \frac{\Delta x}{2\tau}\left(h_{ij}^{(n)} - h_{i+1,j}^{(n)}\right) \qquad (3)$$

How to compute the SWE's so far:

1. compute required fluxes on the edges

   for h p and q

   for p pv_x and pv_y

   for q qv_x and qv_y

2. compute euler time-step for h, p and q

Source Terms

Pressure Induced Force by Watercolumn

$$\frac{1}{2} g \frac{\partial (h^2)}{\partial x}\bigg|_{ij}^{(n)}$$

h ($\hat{=}$mass of the watercolumn)$\times$ $\frac{h}{2}$ height of the center of gravity $=$ pressure

Pressure Difference due to changes of the bathymetry

$$gh\,\frac{\partial b}{\partial x}\bigg|_{ij}$$

for the same depth over ground a change in the bathymetry induces a pressure difference

Balance of the forces

For a not moving watersurface, the two terms have to be in equilibrium

$$\frac{1}{2} g \frac{\partial (h^2)}{\partial x}\bigg|_{ij}^{(n)} = -gh \left.\frac{\partial b}{\partial x}\right|_{ij}$$

Discretization

regular:

$$h \left.\frac{\partial b}{\partial x}\right|_{ij} = h_{ij} \frac{b_{i+1,j} - b_{i-1,j}}{2\Delta x}$$

Balance of the forces

For a not moving watersurface, the two terms have to be in equilibrium

$$\frac{1}{2} g \frac{\partial (h^2)}{\partial x}\bigg|_{ij}^{(n)} = -gh \left.\frac{\partial b}{\partial x}\right|_{ij}$$

Discretization

regular:

$$h \left.\frac{\partial b}{\partial x}\right|_{ij} = h_{ij} \frac{b_{i+1,j} - b_{i-1,j}}{2\Delta x}$$

stable:

$$h \left.\frac{\partial b}{\partial x}\right|_{ij} = \frac{h_{i+1,j} + h_{i-1,j}}{2} \cdot \frac{b_{i+1,j} - b_{i-1,j}}{2\Delta x}$$

Boundary Conditions

realized by ghost cells

Outflow

- ghost cells are set equal to neihbouring fluid cells

Wall

- normal velocity is set to zero on edge $\rightarrow u_{i-1} = -u_i$
- the other quantities are treated like in for the ghost cells

Optimal Timestep

Velocity of characteristic shallow water wave

$$v_{\mathrm{C}} = \sqrt{gh_{\mathrm{max}}} + \max(u_{\mathrm{max}}, v_{\mathrm{max}})$$

leads a a maximal allowed timestep

$$\tau_{\mathrm{max}} = \frac{\Delta x}{v_{\mathrm{C}}}$$

which is computed after each iteration.

1. compute the maximum time-step
2. set ghost cells for boundary treatment
3. compute bathymetric sources
4. compute fluxes (there are more edges than cells)
5. do Euler time-step

Some Details for the CUDA Implementation

- no domain decomposition required (one big block with ghost cells)
- domain is departed in smaller threadblocks of TILE_SIZE
- each thread computes one cells unknowns
- the respective data structures have to be created on the device
    - h, u, v for each cell $\rightarrow$ `hd,hud,hvd`
    - the fluxes for each edge (`Fhd,Fud,Fvd` for fluxes in x-direction and `Ghd,Gud,Gvd` for fluxes in y-direction)
    - the bathymetry `bd`
    - the bathymetric source `Bxd,Byd`

Structure of the Framework

`TODO` marks places where something has to be added

Program structure

compute Boundary on host, already implemented

bath. sources have to be computed on device and stored in local memory (`Bxd,Byd`)

compute fluxes on device, remember that there are more edges than cells, c-code as comment

Euler time-step on device, c-code as comment

max. time-step on host, already implemented