Geophysical Research Abstracts
Vol. 12, EGU2010-10828, 2010
EGU General Assembly 2010

Three Numerical Methods and Their Parallelization for Dual-Lithology Sedimentation

Wenjie Wei, Stuart R. Clark, Xing Cai, and Are Magnus Bruaset
Center for Biomedical Computing, Simula Research Laboratory, Norway(wenjie@simula.no)

article

Three Numerical Methods and Their Parallelization for Dual-Lithology Sedimentation

Wenjie Wei Stuart R. Clark Xing Cai Are Magnus Bruaset

Center for Biomedical Computing, Simula Research Laboratory, Norway

Mathematical Model

High-resolution stratigraphic basin simulation is a challenging and intensive computational task, due partly to complexly interacting physical processes, and partly to vast spatial areas and long time ranges involved. If topographical diffusion is considered as the main driving force, the following system of two partial differential equations can act as an applicable mathematical model:

$$\frac{\partial h}{\partial t} = \frac{1}{C_s}\nabla\cdot(\alpha s\nabla h) + \frac{1}{C_m}\nabla\cdot(\beta(1-s)\nabla h), \quad (1)$$

$$A\frac{\partial s}{\partial t} + s\frac{\partial h}{\partial t} = \frac{1}{C_s}\nabla\cdot(\alpha s\nabla h), \quad (2)$$

where $h(x,y,t)$ denotes the topographic elevation of the basin, and $s(x,y,t)$ denotes the fractional quantity of sand. We note that the above mathematical model considers two lithologies: sand and mud, i.e., $1-s$ is the fractional quantity of mud. Moreover, $C_s(x,y)$ and $C_m(x,y)$ are prescribed concentration fractions of sand and mud, $\alpha(x,y)$ and $\beta(x,y)$ are the associated diffusion coefficients of the two lithologies, and constant A is the thickness of a surface transport layer. The mathematical model (1)-(2) follows that proposed in Rivenæs[1992].

Three Numerical Methods

Computer basin simulations using the above mathematical model correspond to solving an initial-value problem by time integration. At time step ℓ, the latest numerical solutions of h and s, denoted by $h^{\ell-1}$ and $s^{\ell-1}$, are used as initial values. The time derivatives are always approximated by

$$\frac{\partial h}{\partial t} \approx \frac{h^\ell - h^{\ell-1}}{\Delta t}, \quad \frac{\partial s}{\partial t} \approx \frac{s^\ell - s^{\ell-1}}{\Delta t}.$$

Different discretizations of the right-hand sides of (1)-(2) will give rise to different numerical strategies, which often have different characteristics in accuracy, stability and computational speed. The first objective of the present work is to carefully evaluate the following three numerical methods:

1. *Fully-explicit*. Values of $h^{\ell-1}$ and $s^{\ell-1}$ are used in the spatial discretization of the right-hand side of (1), whereas values of h^ℓ and $s^{\ell-1}$ are used for (2). No solution of linear systems is thus needed.

2. *Semi-implicit*. Values of h^ℓ and $s^{\ell-1}$ are used when discretizing the right-hand side of (1), whereas values of h^ℓ and s^ℓ are used for (2). The two discretized equations are thus also decoupled from each other, but two separate linear systems need to be solved in sequence. To achieve second-order temporal accuracy, the Crank-Nicholson scheme may be adopted when discretizing the right-hand sides, together with a few internal iterations within each time step.

3. *Full-implicit*. Values of h^ℓ and s^ℓ are used when discretizing the right-hand sides of both the equations. The two discretized equations are thus coupled with each other, giving rise to a system of nonlinear algebraic equations, which may be solved by Newton-Raphson iterations. We also note that the linear system inside each Newton-Raphson iteration is twice as large as those in the semi-implicit method.

Parallel Computing

The second objective of the present work is to investigate two parallel programming models—MPI and OpenMP—for programming the three numerical methods. Targeting computer clusters based on multicores, MPI will be used for communication between subdomains, each may occupy an entire compute node or just a single core. As memory is shared between the cores within one compute node, OpenMP is an alternative to MPI for intra-node collaboration between the cores. Therefore, the two parallel programming models will be compared with each other, in respect of programmability, flexibility, scalability and overall performance.