Virtual Dike and Flood Simulator: Parallel distributed computing for flood early warning systems

Melnikova, N.B.; Shirshov, G.S.; Krzhizhanovskaya, V.V.; Shabrov, N.N.

Publication date
2011
Document Version
Final published version
Published in
Параллельные вычислительные технологии (ПаВТ'2011)

Virtual Dike and Flood Simulator: Parallel distributed computing for flood early warning systems*

N.B. Melnikova[1,2,3], G.S. Shirshov[1,3] ,V.V. Krzhizhanovskaya[1,2,3], N.N. Shabrov[1],

State Polytechnic University, St. Petersburg, Russia[1]
National Research University ITMO, St. Petersburg, Russia[2]
University of Amsterdam, The Netherlands[3]

The paper presents two simulation modules –Virtual Dike and Flood Simulator– developed for flood Early Warning Systems (EWS). The *UrbanFlood* EWS is a distributed system that (1) analyzes sensor data received in real-time from flood defenses (dikes, dams, etc.) and (2) simulates dike stability, breaching and flood propagation. Computational modules are invoked by workflow-based expert scenarios via the Common Information Space middleware. The Virtual Dike and Flood Simulator have been ported to the HPC Cloud system of SARA supercomputing centre in Amsterdam. Cloud system provides dynamic resource allocation and remote user control. Virtual Dike is a parallel FSI module for coupled simulation of porous flow and dike deformation, based on the finite element method. Efficiency of the distributed EWS and solver parallel performance are presented.

1. Introduction

Design of Early Warning Systems (EWS) for flood protection and disaster management poses a grand challenge to scientific and engineering communities. Development of an EWS includes [1]:

- Sensor equipment design, installation and technical maintenance in flood defense systems;
- Information and Communication Technologies in application to:
  - gathering, processing and visualizing sensor data;
  - developing Common Information Space (CIS) middleware for connecting sensor data, relevant documents, analysis tools, modeling software and advanced scientific visualization;
  - providing Internet-based interactive access to CIS for researchers;
- Development of computational models and simulation components for analysis of dike stability, evaluation of dike failure probability, prediction of flood dynamics and ways for evacuation;
- Development of a decision support system for public authorities and citizens that will help making informed decisions in case of emergency and in routine dike quality assessment, thus reducing flood risk and providing advanced tools for flood management.

Recent catastrophic floods around the world have spawn a large number of projects aimed at the development of stronger and "smarter" flood protection systems. Many projects, among which are FLOODsite, Flood Control 2015, International Levee Handbook [2], attempt to solve some of the EWS aspects listed above. The *UrbanFlood* EC FP7 project [3,4,5] is the first endeavor that unites the work on all the development aspects of a fully integrated EWS. One of the key challenges that we are concentrated on is the development of computational models. Here we describe two models: Flood Simulator and Virtual Dike.

Structural analysis of dike stability includes analyzing different possible failure mechanisms, such as macro-instability, surface erosion and piping. Macro-instability analysis requires modeling of flow through porous media and deformations in the dike -a highly nonlinear coupled fluid-structure interaction problem. Elastic and elasto-plastic models are commonly used to describe soil behavior [6]. In this work, elastic rheological model has been used for dike stability analysis.

In this paper, we briefly describe the *UrbanFlood* EWS workflow and implementation of the distributed system using HPC Cloud of SARA supercomputing centre. Further, we describe the

*The work is supported by [a] *UrbanFlood* EC FP7 project N 248767, theme ICT-2009.6.4 ICT for Environmental Services and Climate Change Adaptation www.urbanflood.eu;
[b] 'Leading Scientist Program' of the Government of the Russian Federation, under contract 11.G34.31.0019.

Virtual Dike and Flood Simulator computational models and parallel efficiency tests for the structural analysis simulations.

2. *UrbanFlood* early warning system computational workflow

Computational workflow of the EWS is organized as follows (see Figure 1): The 'Sensor Monitoring' component receives sensor data from the sensors installed in the dike. Raw sensor data is filtered by the 'Artificial Intelligence Anomaly Detector' that identifies abnormalities in dike behavior or sensor malfunctions. The 'Reliability Analysis' module calculates the probability of dike failure in case of abnormally high water levels or an upcoming storm and extreme rainfalls. If the failure probability is high then the 'Breach Simulator' predicts the dynamics of a possible dike failure, calculates the water discharge through the breach and estimates the total time of the flood. After that, the 'Flood simulator' models the inundation dynamics. Information from all the modules is visualized in the 'Decision Support System', which works as an intelligent interactive interface from the raw data to the users.

For the advanced research into dike stability and failure mechanisms, the 'Virtual Dike' component is available for experienced users. It provides more fundamental and accurate simulations, but requires substantial computing resources, ranging from a powerful computer cluster to supercomputers in case of 3D fully coupled fluid-structure interaction dynamics. The simulation modules and visualization components are integrated into Common Information Space. They are accessed from the interactive graphical environment of the multi-touch table or the web-based application.

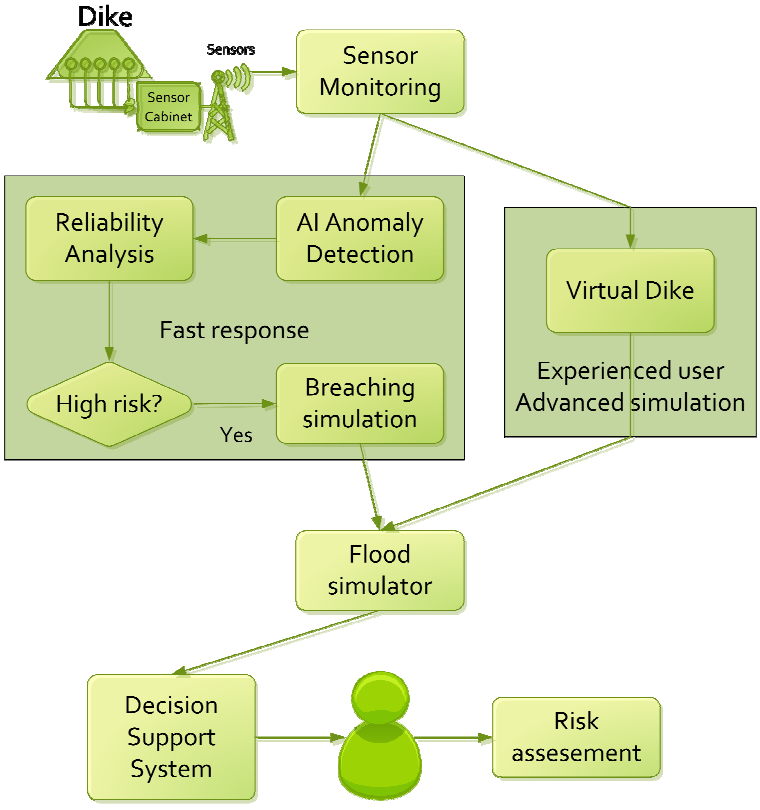

Figure 1. *UrbanFlood* early warning system workflow

3. Implementation of the distributed EWS and cloud computing

The EWS components have been wrapped as plug-ins and integrated in the *UrbanFlood* Common Information Space (CIS). CIS is a generic framework for creating and hosting EWS systems. CIS is used for:

- Monitoring: data is collected in real time from sensors and fed to an EWS.
- Analysis: one or more EWS-specific applications/workflows are invoked to perform analysis of sensor data streams, such as anomaly detection or simulation based on expert models.
- Value judgment: the outcome of this analysis is judged according to EWS-specific rules in order to estimate the risk level or determine the necessity of taking action.
- Advice/act: if mandated by the value judgment, further EWS-specific applications/workflows may be invoked in order to recommend actions for users interacting with a Decision Support System (DSS), or automatically take action to influence the system.

CIS runs on the integration platform PlatIn by *UrbanFlood*. PlatIn extends the engines and introduces additional capabilities which exploit the cloud environment managed by the Dynamic Resource Allocation (DyReAlla) module.

The EWS components Flood Simulator and Virtual Dike have been deployed on the clouds of the SARA BiG Grid High Performance Computing and e-Science Support Center [7]. The cloud is hosted on a 128-core cluster with the following characteristics: 16 compute nodes; CPU dual quad-core 2.2 GHz; 24 GB RAM per node; 500 GB local hard disk; 100 TB backup storage, network 1 Gb/s per node; 20 Gb/s aggregated connection from the cluster to storage. SARA uses OpenNebula open source cloud computing management toolkit. The Virtual machine software is KVM. Simulations are run under Ubuntu Linux.

4. Virtual Dike computational module

Virtual Dike is an advanced multiscale multi-model simulation lab for expert users and model developers. This virtual lab is used for validation of all the models involved in the modeling cascade, and serves as a research field for experiment planning and understanding the underlying physical processes influencing dike stability and failure. Comparison of simulation results with the experimental data allows determining the material properties and computational model parameters that best represent real-life dikes, with all their inhomogeneities and special features. In the first stage of the project, we have studied the structural stability of the LiveDike, a sea dike in Eemshaven. LiveDike is protecting a seaport in Groningen. This dike has been equipped with sensors, and data stream is available in real-time via the LiveDike Dashboard [8]. Pore pressure and dike inclination sensors are placed in four dike cross-sections. These cross-sections have been simulated in 2D models under tidal water loading. Simulation results have been compared with the pore pressure sensors data in order to calibrate soil properties, so that virtual and real sensors agree.

4.1 Modeling approach

The problem requires modeling of coupled fluid-structure interaction with non-linear dike material properties. The fluid part of the model describes the dynamics of flow through porous soil, with Richards equation for wetting and drying of the area above phreatic surface. Van Genuchten model [9] is used to describe the properties of unsaturated soil. As pores' volume expand with the expansion of the media, water content increases. This process is taken into consideration with additional source term in the right hand side of the Richard's equation:

$$(C + \theta_e S)\frac{\partial p}{\partial t} + \nabla \cdot [-\frac{K_S}{\mu} k_r \nabla (p + \rho g z)] = \frac{\partial \varepsilon}{\partial t}, \quad (1)$$

where p is pressure, [Pa]; $C = C(p)$ is specific moisture capacity [1/Pa], given by formula $C = \partial\theta/\partial p$; $\theta_e = \theta_e(p)$ is effective water content; S is a storage coefficient, [1/Pa]; t is time, [s]; μ is dynamic viscosity of water, [Pa·s]; K_S is saturated permeability, [m^2]; $k_r = k_r(p)$ is relative permeability, given

by Van Genuchten formulas; g is standard gravity, [m/s^2]; ρ is water density [kg/m3]; z stands for coordinate of vertical elevation, [m]; ε is volume expansion coefficient.

The structural part of the model describes deformation dynamics of the dike under tidal pressure load, gravity and volumetric pore pressure load obtained from flow simulation. Linear elastic constitutive relation is used for describing sand and clay properties:

$$\begin{cases} \nabla \cdot (\underline{\sigma} - p\underline{E}) + \rho_s \underline{g} = 0 \\ \underline{\sigma} = \lambda \varepsilon \underline{E} + 2\mu \underline{\varepsilon} \end{cases}, \quad (2)$$

Dike stability is evaluated by the Mohr-Coulomb failure criterion [10]. Harmonic approximation of water level sensor signal is applied as input parameter for the flow boundary conditions (to define pressure level at the sea side). At the land side, water stays at the constant average see level. The remaining boundaries are treated as impermeable. In the structural task, corresponding transient pressure is applied at the sea side and constant pressure below average sea level at the land side. The base of the dike is fixed. The remaining boundaries are free from structural loads.

4.2 Implementation and simulation results

Partial differential equations (1), (2) are solved by the finite element method using a commercial software package Comsol [12]. All the simulations are transient. Implicit time integration scheme is applied. Simulations are non-linear due to non-linear constitutive behavior of the soil and fluid-structure coupling phenomenon. Iterative Newton-Raphson method is used to linearize differential equations at each time step. Direct parallel UMFPACK and PARDISO solvers have been applied for solving system of linear algebraic equations within each iteration.

Simulated pore pressure field and structural displacements (at some chosen moment of time) are shown in Figure 2 and Figure 3, correspondingly. Pressure distribution is almost hydrostatic. Pressure changes with elevation mostly. The structural displacements are composed of: a) static soil settlement under gravity load (maximal at the top of the dike) and b) transient displacements resulting from tidal pressure at the seaside and volume pore pressure load. Total displacements are maximal at the top of the dike due to gravity settlement component.

Comparison of pore pressure dynamics for real and virtual sensors corresponding to section #1 of the dike is presented in Figure 4. Real sensor signals are shown with bold lines (left plot – sensor 1E4, right plot – sensor 1G2). “Virtual” sensor signals (obtained from simulation) are shown with thin lines. Comparison of the data for sensor 1G2 shows that the real sensor is placed in a less permeable zone than the zone of the virtual sensor is. In virtual model, permeability values will be corrected and G2 sensors will be surrounded with less permeable soil. Some local inhomogeneities of the soil must be taken into consideration in further simulations.

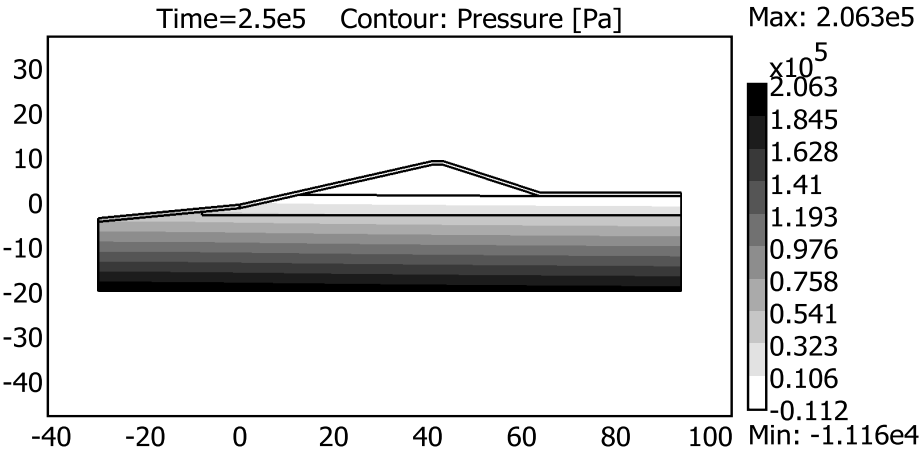

Figure 2. Pore pressure field in the dike

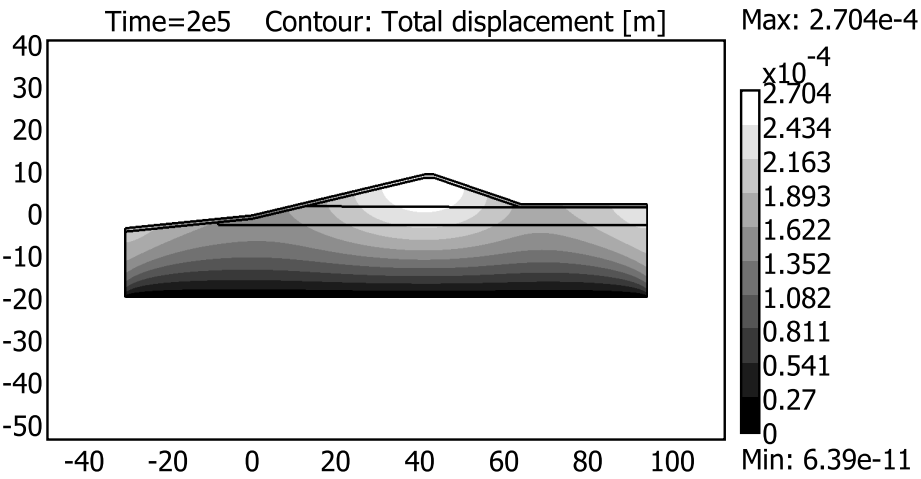

Figure 3. Structural displacement field in the dike

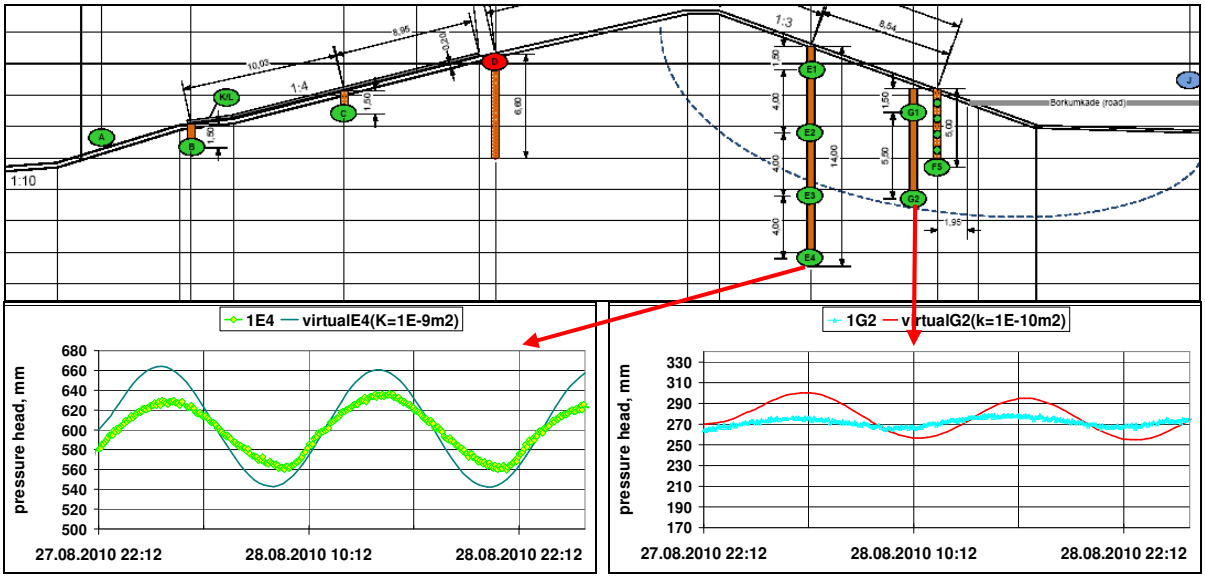

Figure 4. Comparison of pore pressure dynamics for real sensors and virtual sensors

5. Flood Simulator computational module

Flood Simulator module predicts flood dynamics. Given a set of points where a dike breach has occurred, and given the amount of water which will flow through each of these points, this model calculates the water heights of flooded areas over time. The model comprises two main components: a pre-process and a hydraulic simulation.

The objective of the pre-process is to generate the computational mesh which consists of a series of Impact Zones that are based on the floodplain topography and therefore typically irregular in shape. Inputs to this process are the floodplain topography in the form of a Digital Terrain Map (DTM). Creation of the calculation mesh speeds up the process of the simulation (regarding to the DTM data usage).This approach could be used in probabilistic flood risk analysis where multiple runs are required, or in real time situations (flood forecasting), where the model run time is critical.

A hydraulic simulation algorithm performs the following operations at each time step:

1. Boundary Conditions treatment
2. Sort Impact Zones on water level
3. Loop on every wet Impact Zone
   3.1. Calculate outlet discharge towards neighbours (using Manning or weir equations).
   3.2. Discharge limiters

3.3. Use inlet discharge from neighbours
3.4. Transfer volumes
3.5. Calculate velocity

4. Treatment of Impact Zones newly flooded
5. Update water level (volume/level curve) for all wet Impact Zones

Input data is specified in the form of a hydrograph representing a time dependence of water flow rate through the breach in the dike calculated by another simulation module (see Figure 5).

The output of the flood simulator is the table which contains the time series of depth and velocity in each Impact Zone (the saving time-step is defined by the user independently of the computational time-step). The output data is visualized. In the Figure 6 the flooding of the Science Park area of Amsterdam is shown.

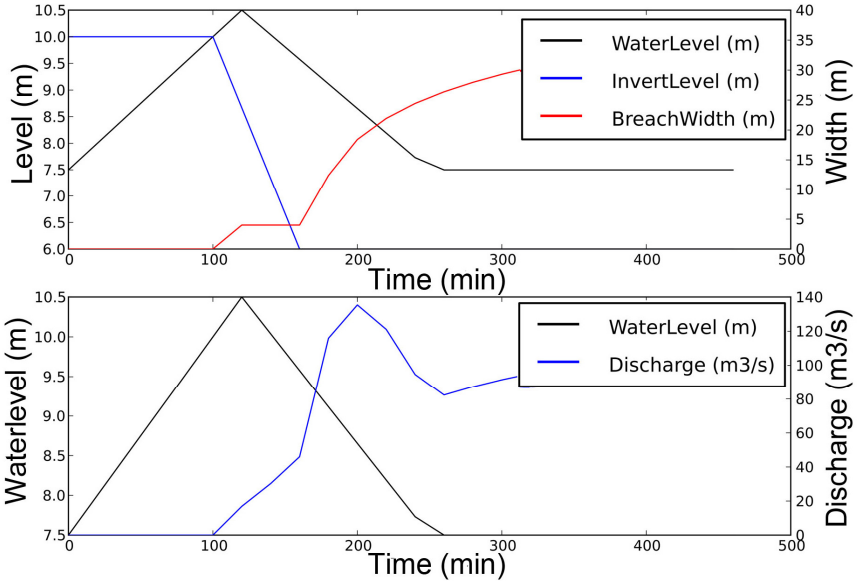

Figure 5. Time dependence of breach width and water amount coming through the breach

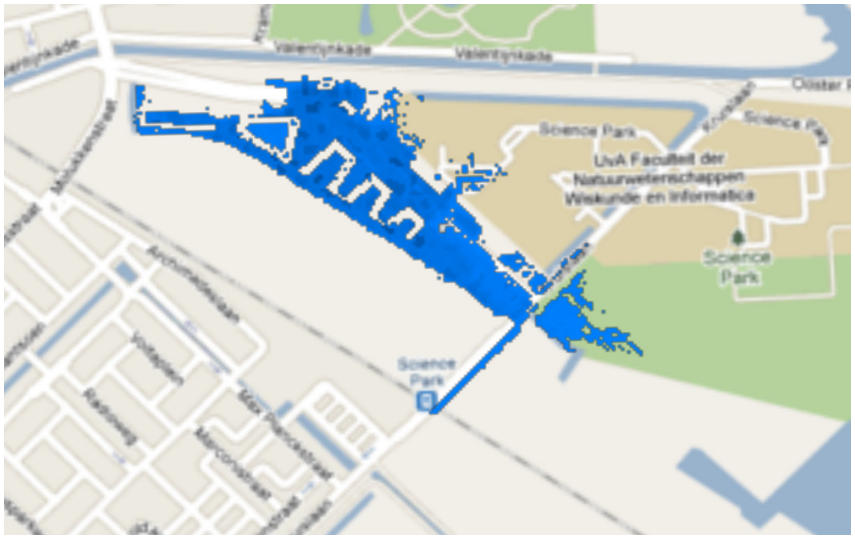

Figure 6. Inundation of the Science Park area - visualization

6. Performance tests of the distributed EWS system and parallel modules

The EWS prototype has been extensively tested and presented during the *UrbanFlood* Workshop [11]. Different EWS components are running on several servers located in 4 countries: Russia, the Netherlands, Poland and the United Kingdom. The modules communicate via the Common Information Space middleware. A user interface implemented on the Microsoft *Surface* multi-touch table provides access to all EWS components and allows interactive simulation steering. Tests of the EWS performance showed a real-time response of system components to user manipulations.

6.1 Parallel efficiency of the Virtual Dike module

For a relatively coarse mesh with 60 000 degrees of freedom (DOF) and maximum element size of 1 m, 2D porous flow simulations typically require up to 2 GB RAM in serial mode. Coupled fluid-structure problem requires up to 8 GB of memory in serial mode. The amount of occupied system memory increases when using parallel mode.

Comsol supports two modes of parallel operation: distributed mode and shared memory mode [12]. Distributed mode employs MPI library for process communications; it can be used on several nodes of a Linux or Windows cluster. Shared memory mode uses shared address space for inter-process communication. It can only be used on a single multi-core or multi-processor computational node. Presently one simulation uses only one computational node of SARA HPC Cloud, with dual quad-core CPU. Shared-memory parallelism has been tested for porous flow uncoupled modeling, employing UMFPACK and PARDISO solvers. Shared-memory parallelism has been used with the default processor usage mode. Besides the default mode, Comsol allows three specific modes to control processor usage: throughput mode (when several different processes are running actively at the same time), turnaround mode and owner mode (both are recommended for usage when no other processes than Comsol are active). We plan to study the efficiency of these modes in future work.

Parallel performance results are presented in Figure 7. PARDISO solver is recommended by Comsol as providing the best parallel efficiency in comparison with the other solvers. The results confirm this only for “small” problem size with number of DOF=60 000. For larger problem sizes UMFPACK scalability was better. At the same time, UMFPACK speed itself was significantly lower for large problem sizes (see Table 1). Hence, for large problems, it is preferable to use PARDISO as it is almost 50% faster.

Computational time varies from 30 min to 7 hours (Table 1), depending on the problem size, number of cores and chosen solver. The parallel speedup of a particular solver first improves as the number of DOF grows (as computational work portion grows higher relative to the portion of information exchange). After that, we get speedup saturation or even decrease for number of cores np equal to 8. The reason of low scalability and cases of decrease in speedup for np=8 may lie [a]: in high

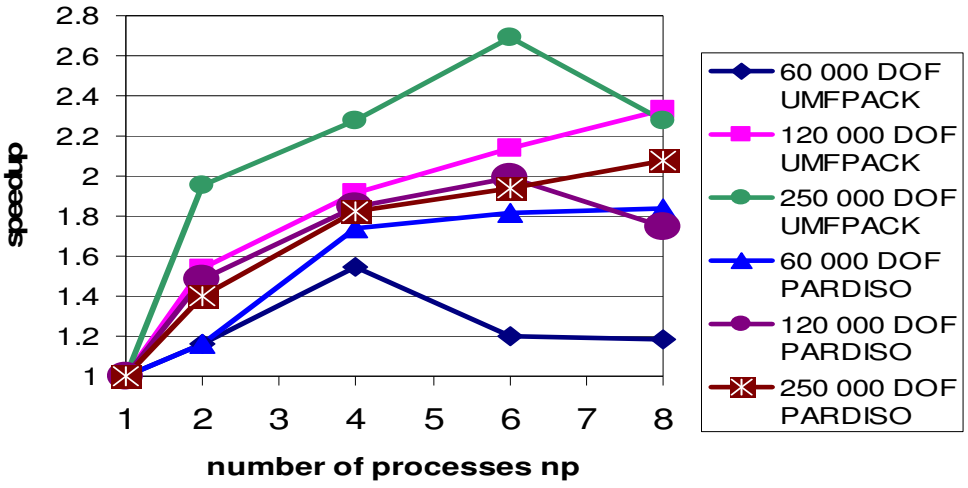

Figure 7. Virtual Dike module: Parallel performance test results

Table 1. Virtual Dike module: parallel simulation time, hours

number of cores	DOF 60000		DOF 120000		DOF 250000	
	UMFPACK	PARDISO	UMFPACK	PARDISO	UMFPACK	PARDISO
1	0.90	0.99	2.30	2.01	7.11	4.93
2	0.78	0.86	1.50	1.35	3.64	3.53
4	0.58	0.57	1.20	1.09	3.12	2.71
6	0.75	0.55	1.08	1.01	2.64	2.54
8	0.76	0.54	0.99	1.15	3.12	2.37

process synchronization costs and [b]: in a relatively high portion of sequential operations included in the algorithm. The solver writes down solution on the hard disk with specified periodicity. As the number of DOF grows, this sequential output work increases. Of course this I/O option could be withdrawn from the simulation tests but we were interested in a real speedup which would happen in working conditions. The output from the solver is going to get to CIS and other computational modules via the hard disk files.

In future research we plan to use a profiler to measure the cost of process synchronization, for different number of cores. A large portion of the computational engine in Comsol relies on BLAS (basic linear algebra subprograms) libraries which are included with the package. By default, Comsol employs MKL (Math Kernel Library) on Intel processors,. This default option can be overridden. An issue for future testing of parallel performance is to explore the efficiency of the alternative BLAS libraries.

7. Conclusions and future work

Design of the EWS for flood protection is in progress. System middleware and basic analysis tools of the EWS have been implemented. Overall efficiency of the distributed EWS has proved to be good. The system provides dynamic resource allocation and controls task scheduling.

We have built computational modules for modeling porous flow and deformations in the dikes; and simulation of inundation in case of dike failure. Parallel speedup of the structural Comsol solver was not high in our tests. The second problem of using Comsol is high memory requirements, which would demand several nodes of HPC cloud for performing distributed 3D simulations. As the primary aim of the analyses tools is to be able to perform simulation in a real-time, a possible solution is to develop a tailored finite-element code for modeling porous flow and structural displacements.

We plan to integrate the Virtual Dike in the Common Information Space to receive sensor input automatically and to produce real-time simulation results for displaying them graphically on a MultiTouch Table. For structural analysis of the dike, elasto-plastic rheological model will be implemented. We plan to perform 3D analysis. Flood condition is to be investigated, including porous flow modeling and dike macro-stability analysis. Parallel efficiency tests are to be continued in 3D case in “pure” conditions (without I/O operations), for different processor usage modes and different BLAS libraries.

Acknowledgements. This work is supported by the EC FP7 project *UrbanFlood*, grant N 248767, and a grant from the 'Leading Scientist Program' of the Government of the Russian Federation, under contract 11.G34.31.0019. The project is carried out in close collaboration with a number of organizations and individuals (listed in alphabetical order): AlertSolutions, particularly Erik Peters; BiG Grid, advanced ICT research infrastructure for e-Science; Deltares, particularly Andre Koelewijn; HRW Wallingford, particularly Ben Gouldby and Julien Lhomme; IJkDijk Association; Rijkswaterstaat, Ministerie van Verkeer en Waterstaat; SARA Computing and Networking Services, particularly Tom Visser and Floris Sluiter; TNO, particularly Jeroen Broekhuijsen and Erik Langius; UvA IBED-CGE, particularly Lourens Veen; UvA GIS , particularly Studio Guido van Reenen; WaterNet, particularly Rob van Putten; Waterschap Noorderzijlvest, particularly Christiaan Jacobs.

References

1. V.V. Krzhizhanovskaya, G.S. Shirshov, N.B. Melnikova, R.G. Belleman, F.I. Rusadi, B.J. Broekhuijsen, B.P. Gouldby, J. Lhomme, B. Balis, M. Bubak, A.L. Pyayt, I.I. Mokhov, A.V. Ozhigin, B. Lang, R.J. Meijer. *Flood early warning system: design, implementation and computational modules*. Proceedings of the International Conference on Computational Science, ICCS 2011. Procedia Computer Science 2011
2. Flood Control 2015 http://www.floodcontrol2015.com, FLOODsite http://www.floodsite.net, The International Levee Handbook project http://www.leveehandbook.net/about-project.asp
3. *UrbanFlood* EU FP7 project http://www.urbanflood.eu
4. V.V. Krzhizhanovskaya. A roadmap to multiscale modeling of flood defense systems: from sand grain to dike failure and inundation. Proceedings of ASME 2010 Computers and Information in Engineering Conference IDETC/CIE 2010, Montreal, Canada. Paper # DETC2010-28967
5. B. Gouldby, V.V. Krzhizhanovskaya, J. Simm, A.G. Hoekstra. Multiscale modelling in real-time flood forecasting systems: From sand grain to dike failure and inundation. Procedia Computer Science, V. 1, N 1, ICCS 2010, May 2010, p. 809
6. A.R. Koelewejn, M.A.Van. Monitoring of the test on the dike at Bergambacht: design and practice. Proceedings of XIII ECSMGE 2003, Prague, Czech Republic.
7. SARA Computing and Networking Services.
8. Livedike dashboard http://livedijk-www.ict.tno.nl/
9. M.T. van Genuchten. A closed form equation for predicting the hydraulic conductivity of unsaturated soils. Soil Science Society of America Journal 44: 892-898.
10. A. Verruijt. Soil Mechanics. Delft University of Technology, 2001. 315 pages.
11. Joint UrbanFlood & SSG4Env Workshop, Thursday 11 & Friday 12 November 2010 – Amsterdam, the Netherlands http://urbanflood.eu/urbanFloodWorkshop2010.aspx
12. Comsol Installation and Operations Guide http://math.nju.edu.cn/help/mathhpc/doc/comsol/install.pdf