DESIGN OF A GRAPHICAL FRAMEWORK FOR EASY PROTOTYPING OF PLUVIAL FLOODING CELLULAR AUTOMATA ALGORITHMS

Michele Guidolin[1], Andrew Duncan[1], Edward C. Keedwell[1], Slobodan Djordjevic[1] and Dragan A. Savic[1]

[1]Centre for Water Systems, College of Engineering, Mathematics and Physical Sciences , University of Exeter, North Park Road, Exeter, EX4 4QF, United Kingdom

[1]*{M.Guidolin, apd209, E.C.Keedwell, S.Djordjevic, D.Savic}@exeter.ac.uk*

Abstract

Cellular automata (CA) algorithms can be used for quickly describing models of complex systems using simple rules. CADDIES is a new EPSRC and industry-sponsored project that aims to use the computational speed of CA algorithms to produce operationally useful real/near-real time pluvial urban flood models for both 1D-sewer and 2D-surface (dual-drainage) flows.

In this paper, the design of a graphical software framework for the CADDIES project is presented. This is intended to simplify the development, testing and use of CA algorithms, and to facilitate the handling of the peripheral tasks of data management and display; allowing the research users to focus on the central tasks of optimisation of CA models and algorithms themselves

Keywords

Cellular automata, pluvial flooding, GUI, framework, dual drainage, computational efficiency, hydrological model

1. INTRODUCTION

The increasing frequency and severity of urban flooding events has increased the importance of pluvial flood modelling which uses an overland flow model to simulate the flow of water on the ground surface and a sewer network model to describe the flow within the drainage system. The concept of the two part model is generally named ‘dual-drainage’, [1] where the ground surface is represented as the ‘major drainage system’ and the sewer network as the ‘minor system’.

The flow propagation on the surface is commonly simulated by two-dimensional (2D) overland models. However, computation of 2D models using fully hydrodynamic models at sufficiently fine resolution is very expensive and can easily take many hours or days to complete in modern hardware. A large amount of research has been done to improve the computation time for this type of model with the objective of producing a real/near-real time urban flood model for uncertainty/risk assessment. Some of the techniques researched to improve the computational time involve the simplification of the 2D terrain features into 1D surface model with a series of ponds/nodes and flow-path/links [2][3]. These new techniques allow the computation to be completed in less than an hour.

Recently, the ability of cellular automata (CA) algorithms (with simple rules) to simulate the complexity of physical models has been investigated in many studies in order to improve dramatically the computation time of 2D surface water models [4][5]. Since flood models that use CA algorithms do not compute fully hydraulic equations but only the non-iterative operation of the CA rules, their execution time could be in the order of minutes. Furthermore, CA algorithms are well suited to be executed in parallel in modern high performance hardware, thus obtaining a very fast speed of computation.

CADDIES is a new project which aims to use cellular automata to improve the speed and efficiency of dual drainage pluvial flood modelling for both 2D urban surface flow and 1D sewer flow, in order to be used for real/near-real time modelling. However, since defining a complex physical model using simple CA rules is not a trivial process, there is a need for a software tool that allows developers to prototype algorithms rapidly and to compare, test, and analyse the results of new and existing algorithms, without the need to develop common code from scratch..

In this paper, the design of a new graphical framework is presented. The idea is to provide an integrated environment where CA algorithms can be easily developed and run while their setup and results data can be inspected all within the same package. Thus the need to invest valuable time and resources by the developers on creation of input and output code, GIS interface and visualisation tools is reduced. Furthermore, the ability to import GIS data and to use new visualisation tools will allow existing case studies to be easily expanded to include use of CA models.

2. CADDIES FRAMEWORK AIMS

The CADDIES framework addresses three main objectives:

1. To simplify the development, testing and comparison of CA algorithms for dual drainage pluvial flood modelling allowing the highest flexibility possible and the complete customisation of the algorithm by the developer.

2. To transparently accelerate the execution of CA algorithms using modern high performance hardware and techniques, i.e. the developer should write the algorithm for serial computation and then achieve higher performance without adding any extra effort into the development.

3. To provide an application with a graphical user interface (GUI) which can be used to manage the development and the execution of the flood modelling simulations and the data produced can be analysed. The application should be able to manage various input output data formats, to visualise different information such as maps and graphs, and to launch the algorithm executions on local and remote machines in order to speed up computation.

The target users of the framework are not only developers/researchers, but also end-users, since the framework should allow anyone to utilise the future CA flood models developed during CADDIES for real/near-real time urban flood modelling.

Given the main objectives of the framework, its main design features are:

- Highly flexible and customisable; since each flooding algorithm has different requirements in terms of input and output data used as well as output visualisation options, the framework should allow researchers and end users to try different algorithms, various data types and visualisation systems. Thus the framework should implement specific techniques in order to easily connect new functionalities.

- Simple; the framework should be easy to use and deploy in order to facilitate its employment by the target user-base. The GUI should be simple, in order to facilitate its use, as well as flexible, in order to accommodate new visualization techniques.

- Portable and fast; the framework should work on different platforms, i.e. different operating systems (O.S.) and different hardware architectures (CPU type). The target is to be able to easily run the framework, or various parts of the framework, on remote machines; and to use all the available processing power (CPU, GPU) in order to minimise run times.

3. CELLULAR AUTOMATA ALGORITHM DEVELOPMENT

Cellular automata [6] offer a versatile method for deriving reduced computational load for models of complex physical systems [7]. A cellular automaton is a discrete model which is composed of a regular lattice of cells defined by a discrete location and a set of states. The evolution of each cell's state is governed by local transition rules which use the previous state of the cell and the states of the neighbouring cells of the CA.

Various CA algorithms can be quite different between similar implementations and are defined by the transition rules implemented, the type of lattice, cell and neighbourhood used. Thus, it is important to provide a high-degree of flexibility for users developing CA algorithms.

The CADDIES framework is designed to give this flexibility. The diagram of figure 1 shows a possible execution of a CA algorithm using the framework. The algorithm is defined by two entities: CA function and CA options. The CA function contains the code of the transition rules to apply in each cell, while the options of a CA algorithm are: the dimensionality of the problem solved (1D, 2D or combination); the type of cell used

(rectangular, hexagonal, triangular, etc.); the type of lattice or grid implemented (regular, rectilinear, unstructured, etc.); the number of neighbourhood cell used; the type of action to execute on the border cells (fixed value, function, wrap-around values, etc.) and the number and type (e.g. discrete, continuous) of cell states.

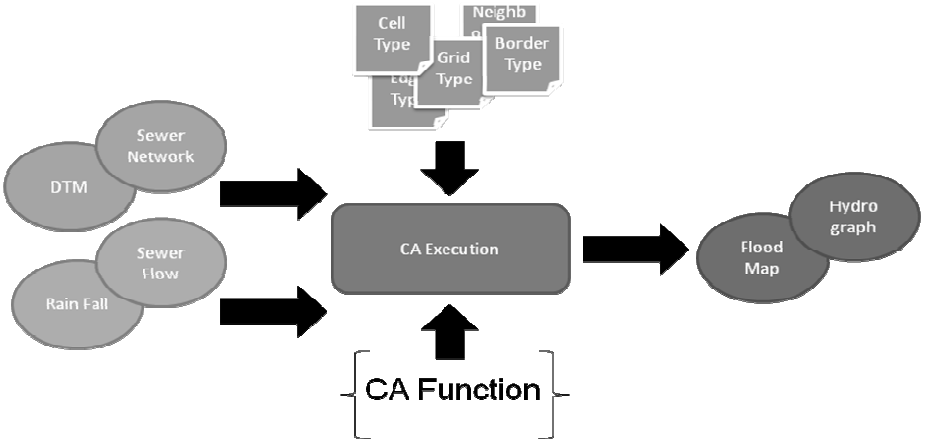

Figure 1. Execution of a CA algorithm with the CADDIES Framework

In the CADDIES framework, the developer will need only to code the CA function and to choose the CA options in order to create a CA algorithm. The creation and management of the cellular lattice, the management of the input and output data, the updating of cell values will all be done automatically by the framework. Thus, the developer will be able to concentrate on implementing the best rules for the flood modelling.

4. AUTOMATIC CA ALGORITHMS ACCELERATION

The idea of the CADDIES framework is that once a developer has implemented the CA rules using the framework and tested the algorithm using sequential computation, he/she should be able to execute the same CA rules using acceleration techniques without changing the code or with minimum effort. There are various high performance techniques that can be used by the framework to speed-up CA algorithms. These techniques can be divided in two types: 1) parallel computation; and 2) minimisation of computation.

Since the cellular automata computations are limited to neighbourhood cells, the local rules are identical for the whole lattice, and the values used in the computations are the ones of the previous step; CA algorithms have proved to have an inherent massively parallel computation capability. Thus, it is possible to run parallel computations of CA algorithms using multi-threading, vector instructions, and the large computational capacity of modern graphics processing units (GPUs) [8][9]. To run any of these parallel executions, the framework will divide the CA lattice on various sub-grids which will be distributed between the various computational units (CPUs or GPUs) available. However, since the various computational units (CUs) do not always have the same computational power, the framework will use dynamic load balancing techniques (DLB) when necessary, i.e. dynamically changing the amount of computations made in each CU depending on its load. There are many DLB techniques available [10], [11]; a common approach between them is to divide the lattice in many more sub-grids than CUs available, assign multiple sub-grids to each CU; and then move the sub-grids between CUs depending on their loads.

Another way to speed-up the execution of a CA algorithm is to minimize the number of computations made. An accelerating technique is to execute pre-processing and post-processing on the lattice, in order to lower its resolution in different areas depending on particular static condition such as height of terrain or distance from river or degree of urbanisation. A more complex technique is to use adaptive mesh refinement (AMR) [12], i.e. dynamically increasing or decreasing the resolution of the cells during the computation depending on the status of the cell. Finally, a way to minimise the computation on a CA algorithm is to compute in each step only the wet-cells and their neighbourhood cells.

Other than implementing these high performance techniques to speed up single CA algorithms, the framework is designed to use distributed computing and cloud computing, i.e. remote executions. This will allow the researches to run in parallel multiple CA algorithms or multiple instances of the same algorithm with different datasets for comparison purposes.

5. FRAMEWORK APPLICATION

The final objective of the CADDIES framework is to develop an application with a graphical user interface which is used to develop and test the CA algorithms and it is used to execute these algorithms for flood risk management. In order to achieve a framework that is useful for both developers and end-users while achieving the objectives set, the application will have two types of workflows: 1) execution-type and 2) execution-run.

The execution-type workflow will allow the researcher to identify the characteristics of the CA algorithm to be executed, the types of input and the types of output that will be used in the flood modelling, and how to visualise the data produced. Thus a researcher, through the use of a specific GUI in the application, will be able to customise all the details of a flood modelling simulation. The execution-run workflow will allow the end-user to execute a flood model of a specific execution-type. Thus, the end-user will be able to choose the data to be used in the execution of the given type and visualise the results using the selected visualisation tools.

5.1 Algorithm evaluation

Once a CA algorithm has been developed is important to evaluate it. Thus, the idea is to provide in the framework a library of metrics that can be used to evaluate the performance of a given CA algorithm: e.g. mass conservation, deviation from real measurement of sensor data, momentum conservation, execution time, etc. The framework will not only provide standard metrics but it will add the option for the users to create, select and apply their own metrics which can be exchanged with other researchers/users.

However, since the development of a new flood model algorithm is made of various different partial implementations; it is important to keep a history of the outputs produced and the metric results of these partial implementations in order to understand which modification is the most effective. Furthermore, it essential to be able to compare the results obtained by a new developed algorithm with the results obtained by existing CA algorithms and existing state-of-the-art models. Thus, the CADDIES framework application is designed to keep a history of each execution-run made. The framework, for each flood modelling simulation executed, will store the input data used and the output results obtained, the details of the machine used in the execution and the relative computational time, and any other details of the execution in a central database.

5.2 Internal structure

CA algorithms for flood modelling differ between each other not only for their internal transition rules and CA options implemented but also for the type and quantity of data input they need and for the type and quantity of output results they produce. Some examples of differences between algorithms are: the use of digital elevation model with or without buildings, the use of ad hoc spot observation rain data (rain gauge) or dynamic-grid data (rain radar), producing hydrographs for few fixed points or any selected points or grid array output, etc. Furthermore, the input data can be retrieved from different tools in many formats and can be visualised in many ways. For example, various GIS systems, on-line web maps, raster or vector files, 2D maps with or without sewer data, 3D maps, etc.

In order to accommodate different needs between different CA algorithms and allowing any future extension; the framework needs to be flexible. To achieve this flexibility, the internal structure of the framework will consist of three layers: visualisation management layer, I/O management layer and execution layer. Each layer contains different components, which can be integrated into the framework through the use of well defined plug-in interfaces. Thus in the future, it will be possible to extend the functionality of the framework. Figure 2 shows the internal structure of the framework with the three layers and their components.

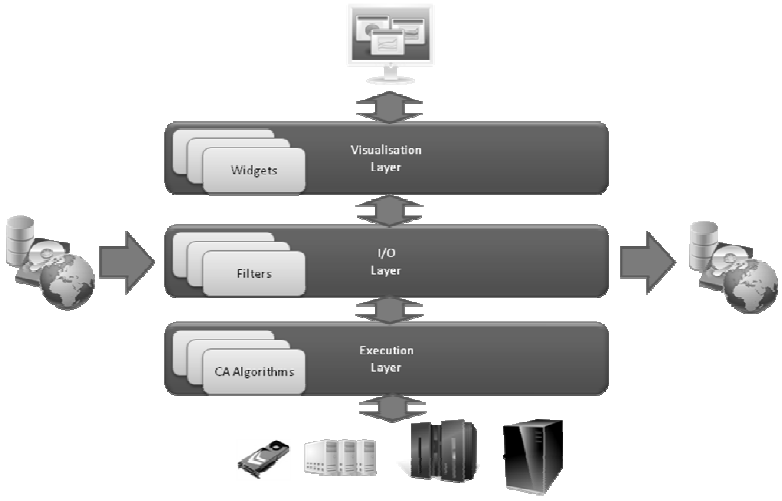

Figure 2. Internal structure of the framework

The I/O layer will contain specific filter components which will be used to read the data from external tools in specific formats and transform the data in the type used internally by the CA algorithms; and vice-versa. The Visualisation management layer will contains the graphical widget components which will be used to read the output results produced by the CA algorithms and visualise the results in human-readable form. The execution layer will contain the CA algorithm components which will execute different CA algorithms in different machines which could be local or remote. Figure 3 shows an example of how the components are used to produce the data input and visualise the output data of CA algorithm of figure 1.

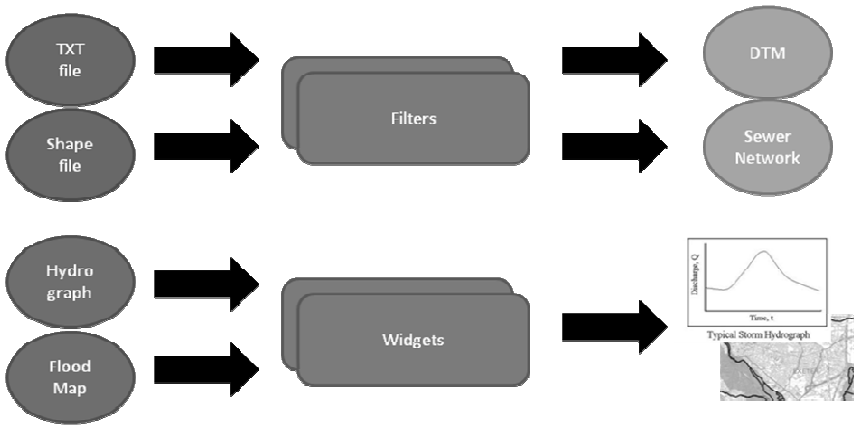

Figure 3.Use of components to read data inputs and visualise data outputs.

Another important design choice made in the CADDIES framework is that the application will be divided in two parts: a backend and a frontend. The frontend will be responsible for the interaction of the user with the framework; thus it will contain the graphical user interface, the tools to read/write input/output data and the visualisation tools. The backend will be responsible for the execution of the various algorithms on the local machine or on the remote machines, thus it will contain a queue of executions. Furthermore, the backend will be responsible for the storage, management and movement of data between machines and the database with the histories of the executions.

6. CONCLUSIONS AND FUTURE WORKS

This work presents the design of framework for rapid development and execution of CA algorithms for dual-drainage pluvial flood model. The design choices made will allow developers to implement, test, compare, and execute cellular automata algorithms for the CADDIES project using a graphical user interface application.

The next step of this work is to start the implementation of the part of the framework that simplifies the cellular automata algorithm's development. Thus, all the CA options available need to be defined in the framework, in addition to the structure of the CA function and the code that creates the lattice and runs the CA rules. Following

this, the next step will be to implement the backend of the application and the plug-in interface. Thus it will be possible to execute the CA algorithm locally and remotely and to create the various filters that read the various input data types. The third step is to design in detail the graphical user interface and to develop the application with the various visualisation widgets. The final step is to introduce the CA acceleration techniques into the framework.

References

[1] Djordjevic S., Prodanovic D., and Maksimovic C., "An approach to simulation of dual drainage," *Water Science and Technology*, vol. 39, pp. 95-103, 1999.

[2] J. Leandro, S. Djordjevic, A. Chen, D. A. Savic, and M. Stanić, "Calibration of a 1D/1D urban flood model using 1D/2D model results in the absence of field data," *Journal of Water Science and Technology*, in press.

[3] J. P. Leitão et al., "Real-time forecasting urban drainage models: full or simplified networks?," *Water Science and Technology: A Journal of the International Association on Water Pollution Research*, vol. 62, no. 9, pp. 2106-2114, 2010.

[4] Y. Liu and G. Pender, "A New Rapid Flood Inundation Model," *Journal of Flood Risk Management*, to appear.

[5] T. J. Coulthard, D. M. Hicks, and M. J. Van De Wiel, "Cellular modelling of river catchments and reaches: Advantages, limitations and prospects," *Geomorphology*, vol. 90, no. 3-4, pp. 192-207, Oct. 2007.

[6] A. Ilachinski, *Cellular automata: a discrete universe*. World Scientific Pub Co Inc, 2001.

[7] S. Wolfram, "Cellular automata as models of complexity," *Nature*, vol. 311, pp. 419-424, Oct. 1984.

[8] J. Tran, D. Jordan, and D. Luebke, "New challenges for cellular automata simulation on the GPU," *SIGGRAPH, Los Angeles. ACM. Poster*, 2004.

[9] S. F. Judice, B. Barcellos, and G. A. Giraldi, "A cellular automata framework for real time fluid animation," in *Proceedings of the Brazilian Symposium on Computer Games and Digital Entertainment*, 2008, pp. 169–176.

[10] M. H. Willebeek-LeMair and A. P. Reeves, "Strategies for dynamic load balancing on highly parallel computers," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 4, no. 9, pp. 979–993, 1993.

[11] D. Cederman and P. Tsigas, "On dynamic load balancing on graphics processors," in *Proceedings of the 23rd ACM SIGGRAPH/EUROGRAPHICS symposium on Graphics hardware*, 2008, pp. 57–64.

[12] M. J. Berger and J. Oliger, "Adaptive mesh refinement for hyperbolic partial differential equations," *Journal of computational Physics*, vol. 53, no. 3, pp. 484–512, 1984.