A MapReduce-Based Multiple Flow Direction Runoff Simulation

Ahmed Sidahmed and Gyozo Gidofalvi

GeoInformatics, Urban Planning and Environment, KTH
Drottning Kristinas väg 30
100 44 Stockholm
Telephone: +46-8-790 8709
Email: {sidahmed, gyozo}@ kth.se

1. Introduction

Using typical Digital Elevation Models (DEM) structures, many algorithms have been proposed, which can be grouped into two groups, single flow direction (SDF), and multiple flow direction (MFD).

In SFD algorithms each cell has only one outlet. The flow is assigned to the neighbor cell with steepest down-slope (Tarboton 1997, Pilesjö and Zhou 1997, Qin et al 2007). The SFD is common and easy to apply (Tarboton 1997), and simple in terms of computation. However, SFD lacks the ability to properly model the flow in flat and relatively high divergent flows areas and creates parallel flows (Qin et al 2007); having only one outlet must be considered as illogical and it imposes great simplification (Pilesjö and Zhou 1997). In the other hand, MFD algorithms distribute the flow among down-slope neighboring pixels proportionally to the slope.

High performance computing was early recognized in the field of hydrological modeling. ParFlow presents a parallel solution for groundwater flows in 3D space (Ashby and Falgout 1996). Tian et al (2008) gave examples of using of Land Information System (developed by NASA) for high-resolution modeling of surface runoff on a high performance computing Linux cluster.

The objective of this paper is to propose and develop a general computational methodology based on the MapReduce framework for grid-based MFD algorithms.

2. BACKGROUND AND RELATED WORK

2.1 MFD

MFD algorithms is proven to be more accurate compared to SFD (Zhou and Liu 2002), but it comes at a high computation cost and complexity as the water in the SFD only flow towards one cell while in MFD water can flow towards eight neighbors. Quinn et al (1991) proposed distributing the flow proportionally to the slope weighted with contour length L, as in Equation 1

$$f_i = \frac{L_i \cdot \tan(\beta_i)}{\sum_{j=1}^{n} L_j \cdot \tan(\beta_j)} \qquad (\tan(\beta) > 0;\ n <= 8) \qquad (1)$$

where f_i represents the fraction of flow towards cell number i, $\tan(f_i)$ and $\tan(f_j)$ is the slope of cell i and j; L_i and L_j is constants of cell i and j; the summation is only for down-

slopes. This method does not consider the degree of divergent formed by the center cell and its eight neighboring cells.

Freeman (1991) proposed similar algorithm based on the slope and a constant *P* as in Equation 2

$$f_i = \frac{\tan(\beta_i)^P}{\sum_{j=1}^{n} \tan(\beta_j)^P} \qquad (\tan(\beta) > 0;\ n <= 8) \qquad (2)$$

and he suggested *P* to be 1.1, which was proved later to not be optimal for all surfaces (Pilesjö and Zhou 1997). Qin et al (2007) argued that this algorithm still does not consider the local terrain conditions. Many Algorithms have been suggested since then; readers are referred to Qin et al (2007) for more discussion on MFD algorithms.

Qin et al (2007) proposed an algorithm that combine Equation 1 and 2, as well as considering the local terrain condition as in Equation 3

$$f_i = \frac{L_i \cdot \tan(\beta_i)^P}{\sum_{j=1}^{n} L_j \cdot \tan(\beta_j)^P} \qquad (\tan(\beta) > 0;\ n <= 8) \qquad (3)$$

the *P* value is determined according to local maximum down slope, this way the algorithm works better for with both divergent and convergent flows. The value of *P* at any given 3 by 3 window can be calculated using the following equation.

$$P = 8.9 \times \min(e,1) + 1.1 \qquad (4)$$

where *e* is maximum local down slope and min(*e*,1) is a function that return the minimum between *e* and 1. This study considered this algorithm to demonstrate the suggested MapReduce approach.

2.2 MapReduce and Hadoop

MapReduce programming framework makes it easily to process in parallel massively large data on large number of computers. MapReduce works mainly through two functions, **Map** function and **Reduce** function; map function takes a set of key/value input and map it into zero or more set of key/value, the reduce function then takes each unique key and group its associated values into a unique key/value set (Dean and Ghemawat, 2008). Hadoop is an implementation of the MapReduce framework.

2.3 MapReduce MFD

The local flow of a cell is defined based on a 3 by 3 neighborhood in terms of possible flow interactions (in and outflow) between the center cell and its neighbors. The global flow can then be achieved by calculating the local flow of each cell by adding the inflow subtract the outflow. The suggested MapReduce can be conceptualized as flowing:

- For each cell form construct a list *l* of nine elements, where the first element represents the center cell and the rest elements represents the eight neighbor cells.
- Each element composed of (*ID;dem;wh*), where *ID* is unique ID for the cell, *dem* and *wh* are DEM and water height respectively at the cell.
- *l* is stored in single line for the purpose of feeding MapReduce.

- The first mapper **Map1** apply MFD algorithm on each *l* and calculate the flow between the center cell and all the neighbor cells with down slope; and the first reducer **Reduce1** aggregate the flows for each cell.
- The second mapper **Map2** writes each cell to the all eight neighbor cells; and the second reducer **Reduce2** reconstruct *l* for each cell.
- Repeating these processes by using the output of **Reduce**2 as an input to **Map1** for each time step.
- The rainfall is added at the first MapReduce job and processed in by the first **Reduce1** as an inflow for each cell.

3. EXPERIMENTS AND RESULTS

3.1 Cluster Setup

Hadoop version 0.20.203 is installed on a cluster of four nodes. Each node has two Intel Xeon E5620 Processor, 1MB L2 Cache, 12M L3 Cache, 2.40 GHz speed, 24GB RAM, one 1Gbps Full Duplex NIC, and 400GB hard drive. Each processor has 4 cores, 8 threads. All nodes are connected using a Gigabit switch. The operating system is Ubuntu 10.04 LTS.

3.2 Dataset

Four mathematics surfaces are used to generate the input gridded DEM. These surfaces are recommended by Zhou et al [8] to evaluate the accuracy of flow algorithms, and it has been chosen as it represents the common train conditions, and in the same time it is easy to generate for the purpose of future comparison.

3.3 Results

To verify the proposed approach two experiments are carried out to execute one time step, cluster and data scaling.

Nodes	Ellips	InvEllips	Saddle	Plane	Ideal
2	0.51	0.48	0.47	0.45	0.50
3	0.40	0.39	0.40	0.37	0.33
4	0.33	0.31	0.30	0.27	0.25

Table 1. Performance for cluster scaling as ratio to single node for the 30 million cells dataset.

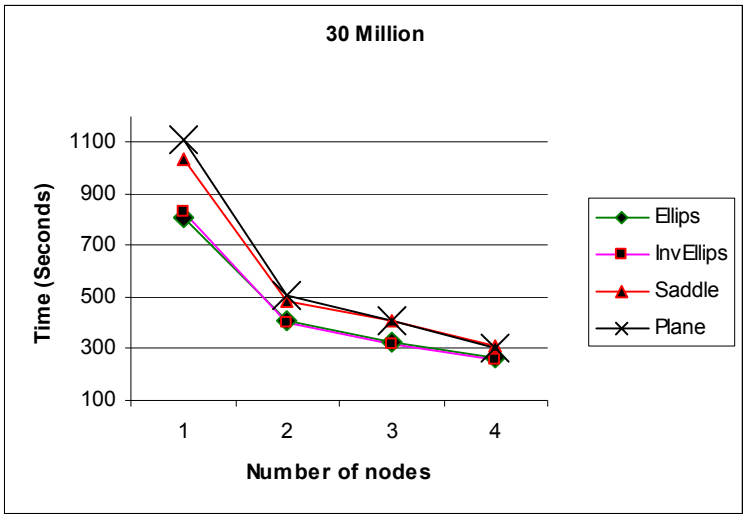

Figure 1. The time compared to the number of server nodes for the 30 million cells dataset.

Figure 1 and Table 1 show the result of the cluster scaling. Fig. 1 shows that the time decrease directly related to the number of nodes for the 30 million dataset. Table 1 compares the achieved time decrease percentage to the theoretically ideal time; however the percentage of two node's time is false indicator because the first node is running as a server worker node in same time.

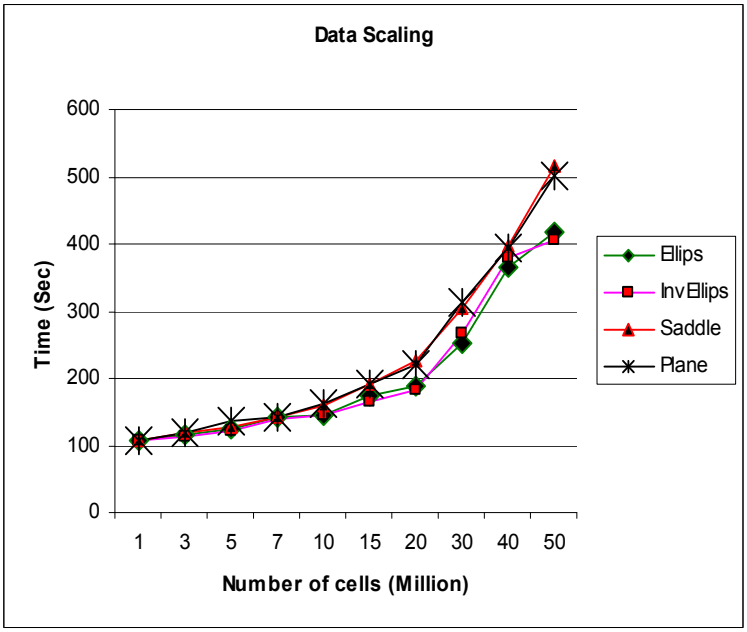

Figure 2. The time of executing one timestamp for different datasets sizes for each surface, on a four node cluster setup.

Fig. 2 shows the execution time for different data sizes on a four node cluster; the relatively flat graph between 1 and 20 million datasets indicates that the cluster is not fully utilized, as opposite to relatively constant increasing from 30 million and more.

The results shows that the suggest approach is suitable for achieving a parallelized MFD on a MapReduce framework, and hence reduced execution time and large amount of data processing ability.

2. References

Ashby, S. F. and R. D. Falgout, 1996, A parallel multigrid preconditioned conjugate gradient algorithm for groundwater flow simulations. *Nucl Sci Eng*, 124:45-59.

Dean, J. and Ghemawat, S, 2008, MapReduce : Simplified Data Processing on Large Clusters. (L. P. Daniel, Ed.) *Communications of the ACM*, 51(1), 1–13. doi:10.1145/1327452.1327492

Freeman, T. G., 1991, Calculating catchment area with divergent flow based on a regular grid. *Computer and Geosciences*, 17:413-422.

Pilesjö, P. and Zhou, Q., 1997, Theoretical estimation of flow accumulation from a gridbased digital elevation model. *Proceedings of GIS AM/FM ASIA'97 and Geoinformatics'97 Conference (Taipei)*, 447-456.

Qin C, Zhu A.-X, Pei T, li B, Zhou C and Yang L, 2007, An adaptive approach to selecting a flow-partition exponent for a multiple-flow-direction algorithm. *International Journal of Geographical Information Science*, 21(4): 443-458.

Quinn P, Beven K, Chevalier P and Planchon O, 1991, The prediction of hillslope flow paths for distributed hydrological modeling using digital terrain models.. *Hydrological Processes*, 5:59-79

Tarboton D.G, 1997, A new method for the determination of flow directions and upslope areas in grid digital elevation models. *Water Resources Research*, 33(2): 309-319.

Tian Y, Peters-Lidard C.D, Kumar S.V, Geiger J, Houser P.R, Eastman J.L, Dirmeyer P, Dotye B and Adams J, 2008, High-performance land surface modeling with a Linux cluster, *Computers & Geosciences*, 34:1492-1504.

ZHOU Q and LIU X, 2002, Error assessment of grid-based flow routing algorithms used in hydrological models. *International Journal of Geographical Information Science*, 16(8): 819-842.