18th World IMACS / MODSIM Congress, Cairns, Australia 13-17 July 2009
http://mssanz.org.au/modsim09

Hydrologic Terrain Processing Using Parallel Computing

Wallis, C. [1], R. Wallace [3], D.G. Tarboton [2], D.W. Watson [1], K.A.T. Schreuders [2], T.K. Tesfa [2]

[1] Computer Science, Utah State University, Logan, Utah, USA
[2] Utah Water Research Laboratory, Utah State University, Logan, Utah, USA
[3] US Army Engineer Research and Development Center, Information Technology Lab, Vicksburg, Mississippi, USA
Email: Robert.M.Wallace@usace.army.mil

Abstract: Topography in the form of Digital Elevation Models (DEMs), is widely used to derive information for the modeling of hydrologic processes. Hydrologic terrain analysis augments the information content of digital elevation data by removing spurious pits, deriving a structured flow field, and calculating surfaces of hydrologic information derived from the flow field. The increasing availability of large terrain datasets with very small ground sample distance (GSD) poses a challenge for existing algorithms that process terrain data to extract this hydrologic information. This paper will describe a parallel algorithm that has been developed to enhance hydrologic terrain pre-processing so that larger datasets can be more efficiently computed. This paper describes a Message Passing Interface (MPI) parallel implementation for Pit Removal. This key functionality is used within the Terrain Analysis Using Digital Elevation Models (TauDEM) package to remove spurious elevation depressions that are an artifact of the raster representation of the terrain. The parallel algorithm works by decomposing the domain into stripes or tiles where each tile is processed by a separate processor. This method also reduces the memory requirements of each processor so that larger size grids can be processed. The parallel pit removal algorithm is adapted from the method of Planchon and Darboux that starts from a large elevation then iteratively scans the grid, lowering each grid cell to the maximum of the original elevation or the lowest neighbor. The MPI implementation reconciles elevations along process domain edges after each scan. The parallel pit removal algorithm has replaced a serial implementation that was based on a recursive search to identify the pour point outlet of each pit so that the elevation of grid cells within the pit could be raised to that level. Initial tests indicate that the MPI overhead within the algorithm results in slower run times for small problems but produces significantly improved processing speeds for a large grid using sixteen processors. We have also been able to process grids much larger than were possible using the memory based single processor implementation.

Specifically for a modest size grid of 28 x 10^6 grid cells, the serial fill algorithm (base) required 71 seconds to complete. The parallel implementation using 5 processors required 51 seconds. The parallel algorithm using 16 processors required only 20 seconds. For a much larger grid of 404 x 10^6 grid cells the base algorithm required 1289 seconds to complete. The parallel algorithm using 8 processors required 954 seconds while using 16 processors required 474 seconds.

Keywords: *Terrain analysis, Hydrology, Digital elevation models, Message Passing Interface, Parallel Processing*

1. INTRODUCTION

Digital Elevation Models (DEMs) are data structures representing rectangular grids of terrain data composed of cells arranged as a raster, where each cell holds a floating point value equivalent to the elevation of that geographic point above some base value (usually, sea level) (Wilson and Gallant, 2000). Cells are typically arranged in row-major order when stored in memory, analogous to 2-dimensional data arrays. DEMs are derived from the actual ground surface using a variety of methods including photogrammetry, lidar or interferometery and form the basis from which digital relief maps are produced. As these methods have increased in precision and accuracy, DEMs have gone from 30-100 meter resolutions 5-10 years ago to 1-5 meter resolutions today for many areas within the United States. New data collection devices and lower collection costs will speed this trend for locations throughout the World. As a result of the increased precision and file sizes, many of the hydrologic preprocessing and analysis techniques for coarser resolutions and smaller DEMs become prohibitively time consuming when being applied to high-resolution data.

This paper reports results from a project whose goal is to produce a parallel implementation of the TauDEM suite of terrain analysis functions (http://www.engineering.usu.edu/dtarb/taudem) so as to improve runtime efficiency and provide a capability to run larger problems. The most computationally time consuming function in TauDEM is the pit filling function that is the focus of this paper.

In natural topography, where the surface is sculpted by fluvial processes, pits comprising depressions completely surrounded by higher terrain are rare. However, in digital terrain representations, pits comprised of grid cells surrounded by grid cells of higher elevation occur more commonly due to deficiencies in the digital elevation model production processes and generalization in the representation of terrain (Jenson and Domingue, 1988; Jenson, 1991). Drainage correction is the processes of altering (correcting) the DEM to remove these pits. Care needs to be exercised not to "correct" actual terrain pits, and there are procedures to identify real pits to the algorithm so that they are retained.

Drainage correction is generally the first step in established procedures for developing a flow model and deriving flow related fields that augment the information content in a DEM (Beven and Moore, 1992; Wilson and Gallant, 2000; Tarboton and Ames, 2001; Maidment, 2002). The most common approach to drainage correction is to fill pits. Pit filling was first implemented using methods that identify the region draining to each pit and the lowest point on the boundary, the so called pour point, then raising the elevation of all points within the region to at least the pour point elevation (Jenson and Domingue, 1988). TauDEM presently uses an implementation of this approach that first identifies pits then recursively scans upslope to find the pour point so as to be able to raise the elevation within the pit to that level. More efficient implementations of pit filling have been developed (Planchon and Darboux, 2001; Arge et al., 2003). The Planchon approach fills pits by covering the whole surface with a thick layer of "water". Then, it removes the water in excess, working inwards from the edges. Doing so, the algorithm naturally enters the depressions by their outlet. Furthermore, embedded depressions do not need a special procedure. The algorithm has a time-complexity of $O(N^{1.2})$ and so, can process large DEMs with an acceptable time-cost (Planchon and Darboux, 2001). The Arge et al. (2003) approach relies on input-output efficient algorithms that explicitly manage data placement and movement.

Sometimes errors in grid DEMs due to interpolation result in artificial dams across valleys and pit filling has the effect of raising the elevation of a large number of upstream grid cells. This alters the data at the very valley locations where it is often of greatest importance, for example for evaluating wetness index and runoff contributions from partial contributing areas adjacent to streams. To circumvent this problem, breaching or carving the DEM to correct it to allow drainage has been suggested. The TOPAZ package (Garbrecht and Martz, 1995; Garbrecht and Martz, 1997) breaches these artificial dams using a limited (3 or 4 grid cell) search downstream from the pour point of each pit. Soille et al. (2003) presented a carving procedure that removes pits by creating a descending path from each pit to a point having a lower elevation value. Carving paths are identified by a flooding simulation starting from the river outlets, similar to the Planchon et al. (2001) approach for pit filling, and trace a path that a rising tide of water from outside the domain would take in overtopping the pour point and reaching a pit. Soille (2004) presented an optimal pit removal method that improved further on this approach. Once the carving path to a pit is identified, a trade-off between lowering the terrain along this path, or raising the elevations in the pit is evaluated to minimize the alterations of the original DEM.

Information on the position of existing streams may also be used to guide drainage correction (Callow et al., 2007). Approaches include stream burning (Maidment, 1996) that lowers the elevation of all stream grid cells by a preset amount prior to processing and AGREE (Hellweger, 1997), which uses a raster

representation of the known stream network to lower the landscape across a user specified horizontal buffer distance and depth as well as burning a stream at a selected depth.

With the increase of scope and resolution of DEMs, the process of hydrologically correcting large DEMs has become increasingly difficult to perform on serial processors and in some cases, impossible given today's hardware limitations for single-processor systems. The memory required to store these DEMs is now on the order of gigabytes and is steadily growing. Processing these DEMs on a single machine requires significant amounts of memory and often results in computer thrashing – excessive swapping of data between memory and the hard disk – resulting in unacceptably slow performance.

This paper presents an MPI parallel implementation of the Planchon and Darboux (2001) pit fill algorithm. The remainder of this paper is organized as follows: Section 2 introduces the Parallel implementation of Planchon's Fill algorithm method. Section 3 illustrates the effectiveness of the parallel algorithm on a small clustered computing system. Concluding remarks are presented in Section 4.

2. PARALLEL IMPLEMENTATION OF PLANCHON FILL ALGORITHM

Planchon's algorithm (Planchon and Darboux, 2001) was chosen as a starting point for implementation of a parallel drainage correction procedure because it was known to be more efficient than the existing TauDEM algorithm. In addition to being more efficient, this algorithm also uses an approach based on techniques from image processing that are similar to the carving and optimal pit removal methods published by Soille et al. (2003) and Soille (2004). Our strategy is to first implement the Planchon approach to obtain results identical to the existing TauDEM method, and second to improve the code to provide improved pit removal functionality by incorporating carving and optimal pit removal.

Planchon's algorithm (Algorithm 1) is initialized with a new DEM, *P*, of infinite (or very large) height. Around the borders, the new DEM's elevation is reduced to match the original DEM. Once this is completed, our implementation performs a series of scans continuously through the rest of the DEM. During each scan, a cell searches through all its neighboring cells and determines the lowest neighboring cell. The cell must be as high as or higher than its lowest neighboring cell in order to drain or be flat. If the original elevation is greater than or equal to the lowest neighboring cell's elevation, we set that cell to its original elevation. Otherwise we set the elevation equal to the lowest neighboring cell's elevation. Each scan is a combination scanning left to right or right to left and top to bottom or bottom to top. The procedure rotates through all eight combinations of the way these scans can be performed, because varying the direction from which a pit is approached increases the likelihood of it being resolved quickly reducing the overall number of iterations. We stop scanning after a scan completes without changing any cell in the DEM.

Algorithm 1 Parallel Planchon Fill. *D* denotes the original elevation. *P* denotes the pit filled elevation. *n* denotes lowest neighboring elevation of *i* evaluated on the pit filled elevations, *P*. *Send(data, destination)* sends *data* given in the first argument to the process designated by the second argument. *Recv(buffer, source)* receives data from *source* and stores it in *buffer*.

```
ParallelPlanchonFill(…)
1: PlanchonInitialize( D,P)
2: Do
3:     for all i in P
4:         if D(i) > n
5:             P(i) ← D(i)
6:         else
7:             P(i) ← n
8:     endfor
9:     Send( topRow, rank-1 )
10:    Send( bottomRow, rank+1 )
11:    Recv( rowBelow, rank+1 )
12:    Recv( rowAbove, rank-1 )
13: Until P is not modified
```

In order to implement this procedure for large data sets in parallel, a method must be devised to partition the data across multiple processes. This study implements a striped partitioning scheme where the grid is divided horizontally into p equal parts and mapped to p processes, with any portion of the grid remaining being attached the last divided portion. Each process reads in their assigned portion of the DEM from a file, along with a row of cells directly above and below the assigned portions. Each process is allowed to have access to all neighboring cells without the need of any extra communication between processes. This method of partitioning the data offers some benefits, in particular, each process inherently knows which process contains the neighboring portions of the DEM, and communication can be simplified. The striped partitioning scheme, as opposed to a tiled partitioning scheme (where the DEM is divided vertically as well as horizontally,) requires a greater number of data transfers but fewer distinct communications events (two per processor in a north/south orientation for the striped scheme, versus four in a north/south/east/west configuration for a tiled partitioning scheme). Furthermore, the data for the striped scheme is contiguous in

the input data file, resulting in a faster overall load time. This approach can be a disadvantage for data sets that are pre-tiled (not striped) and must be combined before the pit removal algorithm can proceed.

For the parallel approach, each process provides solutions for a portion of the whole DEM. Each process works on its own portion in the same manner as described. However, after each scan, each process is required to send and receive the new elevations along the borders of their portions. In this way, the work and memory can be distributed over a many processes. Although this approach incurs data movement overhead inherent to domain decomposition, the overall speed increase overcomes this when dealing with large grids.

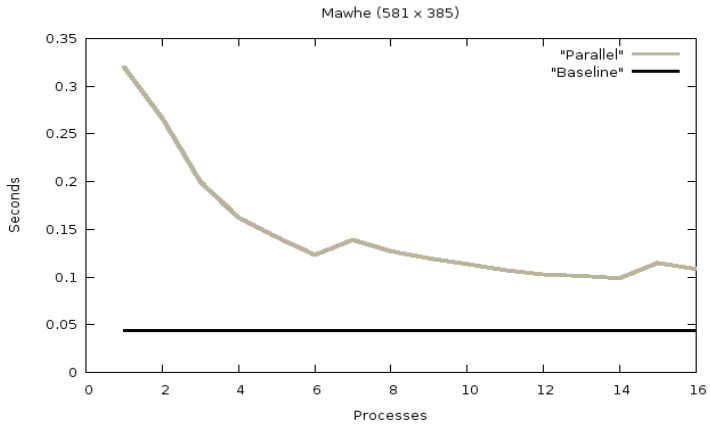

Figure1. Time taken to complete parallel pit fill calculation as a function of the number of processes on a grid of size 581x385, compared to the baseline TauDEM algorithm.

3. EFFECTIVENESS OF THE PARALLEL ALGORITHM

Results for two datasets are included here. The first dataset we used was for a relatively small DEM, so that we could verify the correct functionality of the new algorithm in a parallel environment. In Figure 1 the top line is our parallel Planchon algorithm, compared to the baseline algorithm that is (was) currently used in TauDEM. The fact that the parallel version is slower than the older serial algorithm is not surprising, as the overhead for communication between processes, as well as the different way in which the algorithm progress requires more time than it is worth for this small image.

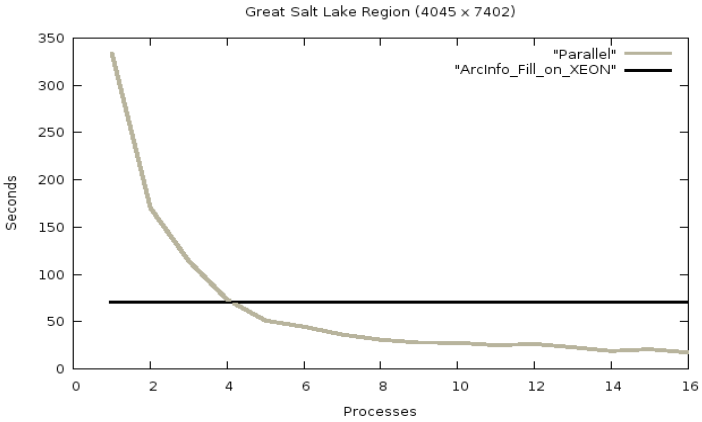

Figure 2. Time taken to complete parallel pit fill calculation as a function of the number of processes on a grid of size 4045 X 7042, compared to ArcInfo fill function on a 3.4 Ghz Pentium Xeon workstation.

In the Figure 2 we present the performance results of our algorithm on a larger DEM. In this test a DEM for the Great Salt Lake area was used. Results using the new algorithm range from 250 seconds for 1 processor to less than 30 seconds for 16 3 GHz Pentium IV processors in the small-scale system in our lab. For comparison purposes, the pit-filling process for this DEM using the old (i.e., non-Planchon) algorithm running on a single-processor system required over 5 days. This comparison includes the speed up due to a better algorithm together with use of parallel processors. Nevertheless, the speed up shown as the number of processors increase is remarkable. We also ran the ArcInfo fill command on the same data set using a 3.4 Ghz Pentium Xeon workstation for this problem. This process required 71 seconds for the serial version to complete. This is in comparison to the 333 seconds needed for a one-processor "parallel" version running on the old Pentium IVs in the cluster. Even with this disparity in execution times for the same task, the parallel version achieved parity with the Xeon processor with 4 processors in the parallel version for this data set. With 16 processors, the pit-filling task was completed in less than 10 seconds.

Finally, we ran the algorithm on a high-resolution DEM entitled 'NedGridb', which is a 14849 x 27174 test dataset, also from the Great Salt Lake region. For comparison, we also ran the Arcinfo pit fill algorithm on the 3.4 Ghz Pentium Xeon, which required 1289 seconds in this case. Only parallel configurations greater

than 8 processors were tested for this dataset, because the machines in the cluster have only 1 gigabyte of memory each. The execution times for this run are given in figure 3.

4. CONCLUSIONS

This paper has presented a parallel algorithm for pit filling of digital elevation models used in the hydrologic analysis of large terrain data sets. The algorithm is based on the original algorithm of Planchon that is relatively straightforward and can be easily augmented for a variety of similar terrain analysis tools. Furthermore, because memory constraints are ameliorated by the increased aggregate memory capacity of cluster systems, the algorithm runs faster than could be anticipated by the linear speedup gains of classic parallel implementations. This algorithm is being incorporated into the parallel implementation of the TauDEM terrain analysis software. We have work under way to incorporate the specification of existing streams in this algorithm by combining a carving approach to force drainage of DEM cells along specified stream paths while using the parallel Planchon Fill to remove pits from the remainder of the DEM. We also have work underway to develop parallel implementations of other TauDEM functions and have achieved similar performance increases for the D8 contributing area calculation (Wallis et al., 2009).

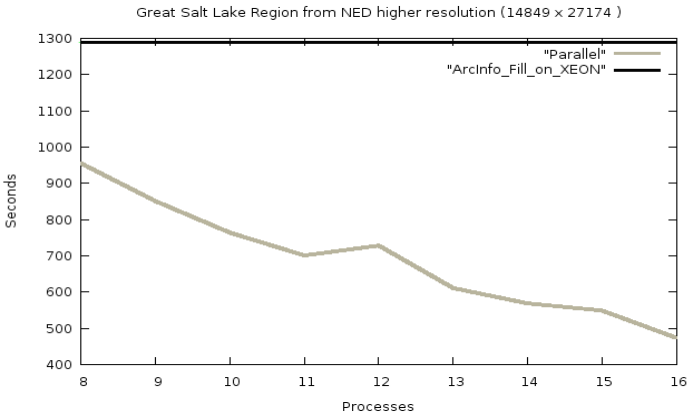

Figure 3. Time taken to complete parallel pit fill calculation as a function of the number of processes on a grid of size 14949 X 27174, compared to ArcInfo fill function on a 3.4 Ghz Pentium Xeon workstation

ACKNOWLEDGMENTS

This research was supported by the US Army Research and Development Center under contract number W9124Z-08-P-0420.

REFERENCES

Arge, L., J. Chase, P. Halpin, L. Toma, J. Vitter, D. Urban and R. Wickremesinghe, (2003), "Efficient flow computation on massive grid terrain datasets," *Geoinformatica*, 7(4): 283-313, http://www.springer.com/geography/gis+cartography/journal/10707.

Beven, K. J. and I. D. Moore, ed. (1992), *Terrain Analysis and Distributed Modeling in Hydrology*, Wiley, 249 p.

Callow, J. N., K. P. Van Niel and G. S. Boggs, (2007), "How does modifying a DEM to reflect known hydrology affect subsequent terrain analysis?," *Journal of Hydrology*, 332(1-2): 30-39, http://dx.doi.org/10.1016/j.jhydrol.2006.06.020.

Garbrecht, J. and L. Martz, (1995), "TOPAZ, An automated Digital Landscape Analysis Tool for Topographic Evaluation, Drainage Identification, Watershed Segmentation, and Subcatchment Parameterization," NAWQL, 95-1, National Agricultural Water Quality Laboratory, USDA, ARS, Durant, OK.

Garbrecht, J. and L. W. Martz, (1997), "The Assignment of Drainage Direction Over Flat Surfaces in Raster Digital Elevation Models," *Journal of Hydrology*, 193: 204-213.

Hellweger, F., (1997), *Agree-DEM surface reconditioning system*. http://www.ce.utexas.edu/prof/maidment/gishydro/ferdi/research/agree/agree.html.

Jenson, S. K., (1991), "Applications of Hydrologic Information Automatically Extracted From Digital Elevation Models," *Hydrological Processes*, 5(1): 31-44.

Jenson, S. K. and J. O. Domingue, (1988), "Extracting Topographic Structure from Digital Elevation Data for Geographic Information System Analysis," *Photogrammetric Engineering and Remote Sensing*, 54(11): 1593-1600.

Maidment, D., (1996), "GIS and hydrological modelling: an assessment of progress," *Third International Conference on GIS and Environmental Modelling*, Santa Fe, NM, 20–25 January.

Maidment, D. R., ed. (2002), *Arc Hydro GIS for Water Resources*, ESRI Press, Redlands, CA, 203 p.

Planchon, O. and F. Darboux, (2001), "A fast, simple and versatile algorithm to fill the depressions of digital elevation models," *Catena*, 46: 159-176.

Soille, P., (2004), "Optimal removal of spurious pits in grid digital elevation models," *Water Resources Research*, 40(12): W12509, doi: 10.1029/2004WR003060.

Soille, P., J. Vogt and R. Colombo, (2003), "Carving and adaptive drainage enforcement of grid digital elevation models," *Water Resources Research*, 39(12): 1366, doi:10.1029/2002WR001879.

Tarboton, D. G. and D. P. Ames, (2001), "Advances in the mapping of flow networks from digital elevation data," *World Water and Environmental Resources Congress*, Orlando, Florida, May 20-24, ASCE, http://www.engineering.usu.edu/dtarb/asce2001.pdf.

Wallis, C., D. Watson, D. G. Tarboton and R. Wallace, (2009), "Parallel Flow-Direction and Contributing Area Calculation for Hydrology Analysis in Digital Elevation Models," *Submitted to PDPTA'09, The 2009 International Conference on Parallel and Distributed Processing Techniques and Applications*, Las Vegas, Nevada, USA, July 13-16.

Wilson, J. P. and J. C. Gallant, (2000), *Terrain Analysis: Principles and Applications*, John Wiley and Sons, New York, 479 p.