Determinação da rede de drenagem em grandes terrenos armazenados em memória externa

Thiago L. Gomes[1], **Salles V. G. Magalhães**[1], **Marcus V. A. Andrade**[1], **and Guilherme C. Pena**[1]

[1] Departamento de Informática – Universidade Federal de Viçosa (UFV)
Campus da UFV – 36.570-000 – Viçosa – MG - Brazil

{thiago.luange,salles,marcus,guilherme.pena}@ufv.br

Abstract. *The drainage network computation is not a trivial task for huge terrains stored in the external memory since, in this case, the time required to access the external memory is much larger than the internal processing time. In this context, this paper presents an efficient algorithm for computing the drainage network in huge terrains where the main idea is to adapt the method RWFlood [Magalhães et al. 2012b] reducing the number of disk access. The proposed method was compared against some classic methods as TerraFlow and r.watershed.seg and, as the tests showed, it was much faster (in some cases, more than 30 times) than both methods.*

Resumo. *A determinação da rede de drenagem é uma aplicação importante em sistemas de informação geográfica e pode requerer um elevado tempo de processamento quando envolve grandes terrenos armazenados em memória externa. Neste contexto, este artigo propõe um método eficiente para computar a rede de drenagem em grandes terrenos, cuja ideia básica é adaptar o método RWFlood [Magalhães et al. 2012b] de modo a reduzir o numero de acessos ao disco. O método proposto foi comparado com outros métodos recentemente apresentados na literatura como TerraFlow e r.watershed.seg e os testes mostraram que o método proposto é mais eficiente (cerca de 30 vezes) que os demais.*

1. Introdução

O avanço da tecnologia do sensoriamento remoto tem produzido um enorme volume de dados sobre a superfície terrestre. O projeto *SRTM (NASA's Shuttle Radar Topography Mission)*, por exemplo, mapeou 80% da superfície da terra com resoluções de 30 metros, formando o mais completo banco de dados de alta resolução da terra, que possui mais de 10 terabytes de dados [Jet Propulsion Laboratory NASA 2012].

Esse enorme volume de dados requer o desenvolvimento (ou adaptação) de algoritmos para o processamento de dados em memória externa (geralmente discos), onde o acesso aos dados é bem mais lento do que na memória interna. Então, os algoritmos para processamento de grande volume de dados (armazenados em memória externa) precisam ser projetados e analisados utilizando um modelo computacional que considera não apenas o uso da CPU mas também o tempo de acesso ao disco.

Uma importante aplicação na área de sistemas de informação geográfica (SIGs) relacionada a modelagem de terrenos é a determinação das estruturas hidrológicas tais como a direção de fluxo, o fluxo acumulado, bacias de acumulação, etc. Essas estruturas são utilizadas no cálculo de atributos do terreno, tais como convergência topográfica, rede de drenagem, bacias hidrográficas,etc.

Este trabalho propõe o método *EMFlow* para a obtenção da rede de drenagem em grandes terrenos representados por matriz de elevação armazenadas em memória secundária. A idéia básica deste novo método é adaptar o algoritmo *RWFlood* [Magalhães et al. 2012b], alterando a forma como os dados em memória externa são acessados. Para isto, é utilizada uma biblioteca que gerencia as transferências de dados entre as memórias interna e externa, buscando diminuir o número de acessos ao disco.

2. Referencial teórico

2.1. Determinação da rede de drenagem

A rede de drenagem é composta pela direção do fluxo de escoamento e pelo fluxo acumulado em cada ponto (célula) do terreno e há diversos métodos para a sua obtenção. Conforme descrito pelos autores, a maior dificuldade neste processo é a ocorrência de *fossos* e *platôs*, ou seja, células onde não é possível determinar a direção de fluxo diretamente porque ou a célula é um mínimo local (fosso) ou pertence a uma região horizontalmente plana (platô).

De acordo com Planchon [Planchon and Darboux 2002], muitos métodos [O'Callaghan and Mark 1984, Jenson and Domingue 1988, Soille and Gratin 1994, Tarboton 1997] utilizam uma etapa de pré-processamento para remover os fossos e platôs e, essa etapa é responsável por mais de 50% do tempo total de execução.

Quando o volume de dados é muito grande e não pode ser totalmente armazenado em memória interna, é necessario realizar o processamento em memória externa e, neste caso, a transferência de informações entre as memórias interna e externa frequentemente domina o tempo de processamento dos algoritmos. Portanto, o projeto e análise de algoritmos utilizados para manipular esses dados precisam ser feito com base em um modelo computacional que avalia o número de operações de entrada e saída (E/S) realizadas.

Vários sistemas de informação geográfica como, por exemplo, o ArcGIS [ESRI 2012] e o GRASS [GRASS Development Team 2010], incluem algoritmos para cálculo da direção de fluxo e do fluxo acumulado. Mas, muitos destes algoritmos são projetados para minimizar o tempo de processamento interno e frequentemente não se ajustam muito bem para grande volume de dados [Arge et al. 2003]. Dentre os métodos desenvolvidos para o tratamento de grande volume de dados em memória externa pode-se destacar os módulos *TerraFlow* [GRASS Development Team 2010] e *r.watershed.seg* [GRASS Development Team 2010] disponíveis no GRASS. O *TerraFlow* é atualmente o sistema que resolve o problema de cálculo de elementos da hidrografia como rede de drenagem e bacia de acumulação (*watershed*) em grandes terrenos de forma mais eficiente [Arge et al. 2003, Toma et al. 2001]. O *r.watershed*, por sua vez, é um módulo do GRASS que pode ser utilizado para a obtenção da rede de drenagem em terrenos e foi adaptado para processamento em memória externa [Metz et al. 2011] com o uso da biblioteca *segmented* do GRASS, que permite a manipulação de grandes matrizes em memória externa.

3. O método *EMFlow*

Em [Magalhães et al. 2012b] é apresentado um novo método chamado ***RWFlood*** que é bem mais eficiente do que os outros algoritmos tradicionais, pois não utiliza uma etapa de pré-processamento para remover os fossos e platôs e os trata naturalmente durante o cálculo da rede de drenagem.

A idéia básica do ***RWFlood*** para obter a rede de drenagem de um terreno é simular o processo de inundação do terreno supondo que a água entra no terreno pela sua borda vindo da parte externa. Neste caso, é importante observar que o caminho que a água percorre à medida que vai inundando o terreno é o mesmo caminho que a água percorreria se fosse proveniente da chuva que cai sobre o terreno e escoa descendentemente.

Mais especificamente, no início, o método cria um oceano em torno do terreno e com nível d'água definido igual à elevação da célula mais baixa entre as células da borda do terreno. Então, é realizado um processo iterativo que, a cada passo, eleva o nível do oceano e inunda as células do terreno. Se a elevação dessas células é menor do que o nível da água então sua elevação é elevada para ficar igual ao nível do oceano.

Inicialmente, a direção de fluxo nas células da borda do terreno é definida apontando para fora do terreno (isto é, indicando que naquelas células a água escoa para fora do terreno). Então, a direção de cada célula c que não pertence à borda é definida como apontando para a célula vizinha a c de onde a água vem para inundar a célula c.

Depois de inundar todas as depressões e todas as células com elevação igual ao nível da água e que são adjacentes à borda do oceano, o nível da água é elevado para a elevação da célula mais baixa que é adjacente à borda desse oceano. Para obter essa célula que irá definir o nível da água, o método ***RWFlood*** utiliza um array Q de filas para armazenar as células que precisam ser posteriormente processadas. Ou seja, Q contém uma fila para cada elevação existente no terreno, sendo que a fila $Q[m]$ armazena as células (a serem processadas) com elevação m. Inicialmente, as células na fronteira do terreno são inseridas na *fila* correspondente. Assim, supondo que a célula mais baixa na borda do terreno tem elevação k, então o processo começa na fila $Q[k]$ (isso corresponde a supor que nível da água se inicia com elevação k). A partir disso, supondo que c é a célula na primeira posição da fila $Q[k]$; essa célula é removida da fila e é processada da seguinte forma: as células vizinhas a c que ainda não foram "visitadas" (isto é, que ainda não têm a direção de fluxo definida) têm a sua direção de fluxo definida apontando para a célula c e elas são inseridas nas respectivas filas. É importante observar que, se uma célula vizinha a c que ainda não foi visitada tem elevação menor do que c, então a elevação dessa célula é incrementada (conceitualmente, isso corresponde a inundar a célula) e depois ela é inserida na fila correspondente a essa nova elevação. Quando todas as células na fila $Q[k]$ são processadas, o processo continua na próxima fila não vazia no vetor Q

Vale ressaltar que o método ***RWFlood*** determina a direção do fluxo de cada célula durante a inundação. Quando uma célula c é processada, todas as células vizinhas a c que ainda não foram visitadas (isto é, que não tem a sua direção de fluxo definida) têm o seu sentido de fluxo definido para c e depois, são inseridas na fila correspondente.

Após o cálculo da direção de fluxo, o algoritmo ***RWFlood*** calcula o fluxo acumulado no terreno utilizando uma estratégia baseada em ordenação topológica. Conceitualmente, a ideia é supor a existência de um grafo onde cada vértice representa uma célula

do terreno e há uma aresta ligando um vértice v a um vértice u se, e somente se, a direção de escoamento de v aponta para u.

3.1. Adaptação do método *RWFlood* para processamento em memória externa

O método *RWFlood* original processa o terreno, representado por uma matriz, acessando essa matriz de forma não sequencial e, portanto, o processamento de grandes terrenos armazenados em memória externa pode não ser eficiente. No entanto, há um padrão de acessos espacial, pois, em um dado momento as células acessadas estão, na maioria das vezes, próximas umas das outras na matriz.

Para diminuir o número de acessos ao disco, este trabalho propõe um novo método denominado *EMFlow*, cuja estratégia consiste em adaptar o método *RWFlood* de forma que os acessos realizados à matriz sejam gerenciados por uma biblioteca denominada *TiledMatrix* [Magalhães et al. 2012a], que é capaz de armazenar e gerenciar grandes matrizes em memória externa. Na verdade, a idéia básica desta adaptação é modificar a forma de gerenciamento da memória (reorganizando a matriz) para tirar proveito da localidade espacial de acesso.

Assim, as matrizes em memória externa são gerenciadas pela biblioteca *TiledMatrix*, que subdivide a matriz em blocos menores que são armazenados de forma sequencial em um arquivo na memória externa, sendo que a transferência destes blocos entre as memórias interna e externa também é gerenciada pela biblioteca que permite a adoção de diferentes políticas de gerenciamento.

Uma questão importante a se considerar na implementação da biblioteca *TiledMatrix* refere-se à política utilizada para determinar qual bloco será escolhido para ceder espaço a novos blocos. Neste trabalho utilizou-se a estratégia de retirar da memória interna aquele bloco que está a mais tempo sem ter sido acessado pela aplicação. Essa estratégia foi adotada baseado no fato de que, durante o processamento do algoritmo *RWFlood*, há uma certa localidade de acesso às células do terreno, assim blocos que estão a muito tempo sem serem acessados tendem a não serem mais acessados. No entanto, serão realizados estudos mais detalhados para verificar se realmente essa é a melhor estratégia.

4. Resultados

O algoritmo *EMFlow* foi implementado em C++, compilado com o g++ 4.5.2, e vários testes foram realizados para avaliar seu tempo de execução e seu comportamento em diferentes situações comparando-o contra os métodos *TerraFlow* e *r.watershed.seg*, ambos incluídos no GRASS. Os testes foram executados em uma máquina com processador Intel Core 2 Duo com 2,8GHz, HD de 5400 RPM e sistema operacional Ubuntu Linux 11.04 64 bits.

Os terrenos utilizados nos testes foram gerados a partir de dados dos EUA disponibilizados pelo projeto SRTM [Jet Propulsion Laboratory NASA 2012] com resolução horizontal de 30 metros.

A tabela 1 exibe os tempos de processamento (em segundos) de uma determinada região utilizando memórias de 1GB e 4GB, sendo que no método *EMFlow* foram utilizados blocos com 200×200 células para a memória de 1GB, e 800×800 para 4GB. No caso do *TerraFlow*, a versão disponível no GRASS utiliza, no máximo, 2GB de memória. No

	EMFlow Tempo(s)		TerraFlow Tempo(s)		r.watershed.seg Tempo(s)	
	Memória		Memória		Memória	
Tamanho	**1GB**	**4GB**	**1GB**	**4GB**	**1GB**	**4GB**
1000^2	0,66	0,81	24,43	19,32	6,36	6,34
5000^2	14,18	15,04	661,37	400,84	625,21	616,53
10000^2	74,56	65,38	2329,71	2251,70	12636,07	8529,70
15000^2	326,15	153,60	7588,33	5870,30	∞	22276,00
20000^2	717,87	295,35	12937,30	13067,00	∞	41493,00
25000^2	2006,14	529,50	22220,89	19340,00	∞	77729,00
30000^2	2848,13	850,53	35408,11	30364,00	∞	∞
40000^2	5653,93	1826,80	67076,04	56421,00	∞	∞
50000^2	10649,04	2897,60	98221,64	82673,00	∞	∞

Tabela 1. Comparação entre os algoritmos de memória externa.

caso *r.watershed.seg*, o símbolo ∞ indica que, naquela situação, a execução do método foi interrompida quando o tempo de processamento ultrapassou 100000 segundos.

Como é possível verificar, o método *EMFlow* apresentou um desempenho bem melhor do que os outros dois métodos em todas as situações, chegando a ser mais de 30 vezes mais rápido. Vale ressaltar que as redes de drenagens produzidas pelo método *EMFlow* são idênticas ao método *RWFlood* [Magalhães et al. 2012b] que, por sua vez, apresenta resultados simulares aos obtidos por ferramentas como ArcGIS [ESRI 2012] e o GRASS [GRASS Development Team 2010].

5. Conclusões e trabalhos futuros

Neste trabalho foi apresentado o algoritmo *EMFlow* para cálculo da rede de drenagem em grandes terrenos armazenados em memória externa e, como mostrado pelos testes, o método proposto apresenta uma eficiência muito superior aos principais métodos disponíveis. Em particular, vale destacar que, em situações extremas (terrenos muito maiores do que a memória interna), o *EMFlow* foi cerca de 30 vezes mais rápido do que o *TerraFlow* e, em muitas dessas situações, não foi possível obter o resultado (num tempo razoável) utilizando o método *r.watershed.seg*.

Um fator importante que afeta a eficiência do método é o tamanho do bloco escolhido na subdivisão da matriz. O próximo passo do trabalho é realizar um estudo mais detalhado de como a escolha desse tamanho pode ser determinado de forma automática.

Ocasionalmente, as filas de processamento do algoritmo podem crescer muito levando a ineficiência dele, para contornar esse problema está sendo avaliada uma forma de dividir o processamento das filas e do terreno de tal forma que cada conjunto não possua relação com o outro, podendo este ser processado separadamente sem qualquer problema.

Agradecimento

Este trabalho foi parcialmente financiado pela CAPES, pela FAPEMIG e pelo CNPq.

Referências

Arge, L., Chase, J. S., Halpin, P., Toma, L., Vitter, J. S., Urban, D., and Wickremesinghe, R. (2003). Efficient flow computation on massive grid terrain datasets. *Geoinformatica*, 7.

ESRI (2012). Arcgis. Disponível em: http://www.esri.com/software/arcgis/arcgis-for-desktop/index.html. (acessado em 17/05/2012).

GRASS Development Team (2010). *Geographic Resources Analysis Support System (GRASS GIS) Software*. Open Source Geospatial Foundation, http://grass.osgeo.org (acessado 17/05/2012).

Jenson, S. and Domingue, J. (1988). Extracting topographic structure from digital elevation data for geographic information system analysis. *Photogrammetric Engineering and Remote Sensing*, 54(11):1593–1600.

Jet Propulsion Laboratory NASA (2012). *NASA Shuttle Radar Topography Mission (SRTM)*. National Geospatial-Intelligence Agency (NGA) and National Aeronautics and Space Administration (NASA), http://srtm.usgs.gov/mission.php(acessado 17/05/2012).

Magalhães, S. V. G., Andrade, M. V. A., Ferreira, C. R., Pena, G. C., Luange, T. G., and Pompermayer, A. M. (2012a). Uma biblioteca para o gerenciamento de grandes matrizes em memória externa. Technical report, Departamento de Informática, Universidade Federal de Viçosa.

Magalhães, S. V. G., Andrade, M. V. A., Franklin, W. R., and Pena, G. C. (2012b). A new method for computing the drainage network based on raising the level of an ocean surrounding the terrain. *15th AGILE International Conference on Geographic Information Science*.

Metz, M., Mitasova, H., and Harmon, R. S. (2011). Efficient extraction of drainage networks from massive, radar-based elevation models with least cost path search. *Hydrology and Earth System Sciences*, 15(2):667–678.

O'Callaghan, J. and Mark, D. (1984). The extraction of drainage networks from digital elevation data. *Computer Vision, Graphics and Image Processing*, 28:328–344.

Planchon, O. and Darboux, F. (2002). A fast, simple and versatile algorithm to fill the depressions of digital elevation models. *Catena*, 46(2-3):159–176.

Soille, P. and Gratin, C. (1994). An efficient algorithm for drainage network extraction on dems. *Journal of Visual Communication and Image Representation*, 5(2):181–189.

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

Toma, L., Wickremesinghe, R., Arge, L., Chase, J. S., Vitter, J. S., Halpin, P. N., and Urban, D. (2001). Flow computation on massive grids. In *GIS 2001 Proceedings of the 9th ACM international symposium on Advances in geographic information systems*.