A Lossless Effective Method for the Digital Elevation Model Compression for Fast Retrieval Problem

Le Hoang Son, Nguyen Duy Linh, Tran Van Huong, and Nguyen Huu Dien*

Hanoi University of Science, Viet Nam National University
334 Nguyen Trai, Thanh Xuan, Ha Noi, Viet Nam

Summary
Digital Terrain Models (DTMs) or Digital Elevation Models (DEMs) have been being used as common ways to construct 3D terrains in GIS. These data can be obtained from topographic maps, aerial surveys, satellite, Doppler radar, etc. However, their sizes are often large depending on the resolutions. Therefore, it is hard to store as well as retrieve them efficiently from DBMS. Besides, some current, striking DEM compression techniques [1], [2], [3], [4], [20], [27] have limitations on compressed time and most of them do not support for retrieval on compressed data. Indeed, these limitations are major obstacles when deploying 3D WebGIS applications over the Internet environment which requires fast processing. In this paper, we will concentrate on the DEM compression for fast retrieval problem and introduce a novel lossless method based on adaptive sliding windows and parallel computation techniques. In fact, our algorithm is an amelioration of David and Derek method [3] which is known as the fastest compression technique of compressed time on DEM data among all DEM compression algorithms [1], [2], [3], [4], [20], [27]. Additionally, this algorithm is evaluated and compared with some best known ones to show its efficiency and suitability for the original problem.

Key words:
DEM compression, Lagrange methods, Parallel Computation, Sliding Windows

1. Introduction

3D Geographical Information System (GIS) is being used as a useful visualization and analysis tool for managers, researchers, etc. It provides a realistic interface that closely relates to what exists on the Earth. Input of 3D GIS is *Digital Elevation Model* (DEM), or sometimes *Digital Terrain Model* (DTM). A DEM can be represented as a raster (a grid of squares) or as a triangular irregular network (TIN). DEMs are commonly built using remote sensing techniques such as photogrammetry, LiDAR, IfSAR, but they may also be built from land surveying, etc [12]. While a Digital Surface Model (DSM) may be useful for landscape modeling, city modeling and visualization applications, a DEM is often required for flood or drainage modeling, land-use studies, geological applications, and much more.

Manuscript received June 5, 2011
Manuscript revised June 20, 2011

Gridded DEM [17] is one of the most widely available forms of environmental data. It is stored as a two dimension array and each cell contains an elevation value. There are different interpretations of the grid. First, the elevation can represent the elevation of every point inside the square. In this case, the DEM is a non-continuous function. Second, the elevation is only belonged to the center point of the square, or is the average elevation inside the square. In this case, some interpolation methods are required to get elevations of all points in the square. Although there are still some arguments about the use of elevation of gridded DEM, this kind of terrain data is undoubtedly recognized as the most popular way to construct 3D GIS terrains in comparison with the rests. Moreover, it has been being used in various applications such as terrain synthesis [6], upslope areas [8], stream network [9], Slope-Extraction Analysis [10], Digital Channel Network Extraction [24], and many other ones [5], [11], [19], [23], [25] , [26].

However, size of Gridded DEM is often large due to its resolution. As stated in [7], the accuracy of DEM and DEM-derived products depends on several factors, including the horizontal resolution and vertical precision at which the elevation data are represented, and the source of the elevation data. This accuracy becomes increasingly important as we extend the use of DEM data for spatial prediction. Indeed, more accuracy leads to more resolution and the size of DEM is increased as a result. Thus, it is hard to store and retrieve DEM data from DBMS without compression.

Consider the following scenario: assume that we have a 3D WebGIS system. A user accesses the official homepage and uses 3D WebGIS to view his terrain data. Then, the user wants to save DEM data to the DBMS database on server for later uses. The server is required to use an efficient spatial compression technique to save the large DEM data into the DBMS database with smallest size if possible. Moreover, these compressed data do not need to be fully extracted for display. This means parts of them are required to be extracted instead of the whole compressed DEM terrain. Such a spatial compression technique, if

well-deployed, is a good choice for fast displaying and storing terrain data in the WWW environment.

The scenario above is our long term purpose and what we want to reach in this paper. Moreover, it is a development of our works on creating a 'smart' 3D WebGIS system [13], [14], [15], [16]. In fact, these have been some DEM compression techniques [1], [2], [3], [4], [20], [27] dealing with this problem. However, most of them have high compressed time and do not support for retrieval on compressed data.

Originated from the results of David and Derek method [3] about fast DEM compression using Lagrange methods, we utilize them and present an amelioration based on the concept of adaptive sliding windows and parallel computation techniques for the DEM compression for fast retrieval problem. Our method is lossless and named as DCR. Certainly, it will be tested by numerical experiments to show the efficiency when comparing with some state-of-the-art algorithms.

The rest of this paper is organized as follows. Section 2 describes the problem in details. Some related works are shown in Section 3. The DCR method is presented in Section 4. Section 5 presents some experiments. Finally, conclusions and future works are described in the last section.

2. Problem Statement

We start with some definitions

- *Def 1*: A *sliding window* (SW) with the original point (x_0, y_0) and sizes (w, h) is defined as

$$SW(x_0, y_0) = \{(x_0 + u, y_0 + v) \mid u \in [0, w]; v \in [0, h]\} \quad . (1)$$

- *Def 2*: *A set of moving steps* for the point (x, y) with arguments (i, j) is denoted

$$\Delta_{i,j}(x, y) = \{(x + \delta_1, y + \delta_2) \mid \delta_1 = -i, 0, i; \delta_2 = -j, 0, j\} \quad . (2)$$

This set contains 8 moving steps $\Delta^k_{i,j}(x, y)$ with $k = \overline{1,8}$. For example

$$\Delta^1_{i,j}(x, y) = (x + i, y) \in \Delta_{i,j}(x, y) \quad . (3)$$

- *Def 3*: The sliding window at the next point after using a specific moving step can be extracted from two previous definitions as

$$SW(x_1, y_1) = SW(x_0, y_0) \mid \Delta^k_{i,j}(x_0, y_0)$$

$$= \{(x_0 + u + \delta_1, y_0 + v + \delta_2) \mid u \in [0, w], v \in [0, h], \delta_1, \delta_2 = \gamma(k)\} \quad , (4)$$

With $\gamma(k)$ are the arguments at k^{th} moving step.

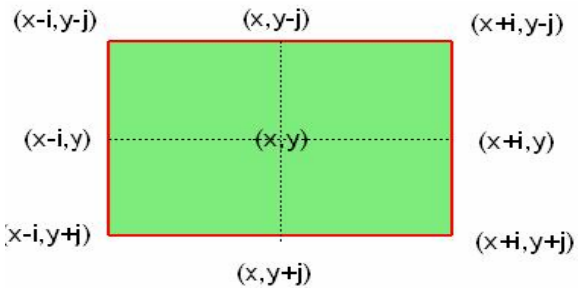

Fig. 1 A set of moving steps.

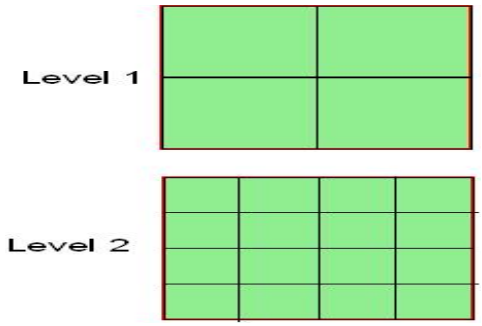

Fig. 2 Two levels of moving steps.

- *Def 4*: (*Level of moving step*)

Without loss of generality, we can assume that sizes of DEM terrain are power of two. In other cases, some rows or columns of specific values can be appended to the terrain for this purpose. Then, the level of moving step is specified as follow. At level *k*:

$$(i, j) = (\frac{W_0}{2^k}, \frac{H_0}{2^k}) \quad , (5)$$

$$(w, h) = (\frac{W_0}{2^k}, \frac{H_0}{2^k})$$

With (W_0, H_0) are sizes of the original DEM terrain. Two couple of parameters (i, j) and (w, h) are described in *Def* 1 and *Def* 2.

- *Def 5*: (*Acceptance range of moving steps*)

The acceptance range of moving step is the range $[K_0, K_{\max}]$ where K_0 is the minimal level of moving steps that makes all sliding windows in this level be

quickly displayed by any web browser. $K_{\max}$ is the maximal level for a given DEM terrain.

$$K_0 = \left| \log_{\frac{1}{4}} (\frac{x_1}{X}) \right| \quad , (6)$$

$$K_{\max} = \min(n) \text{ where } 2^n \geq \max(W_0, H_0) \ .$$

In (6), *X* is the volume or capacity of the original DEM terrain in Megabyte (MB) and x_1 is the specific volume of the maximal sliding window in this level that can be quickly displayed by web browsers and also measured in MB. For example, in JSG framework [16], $x_1 = 1.2$. Here we assume that K_0 is always smaller than $K_{\max}$. Otherwise, re-adjust K_0.

From now on, we denote $SW^k(x_0, y_0)$ as a sliding window with the original point (x_0, y_0) at level *k*.

The *DEM compression for fast retrieval* (DCR) problem can be understood as follow. Suppose that we have a DEM terrain. Our purpose is to compress it for the reduction of terrain data and fast display form a sliding window $SW^k(x_0, y_0)$ at a specific level of moving steps $k \geq K_0$ without extracting the whole compressed terrain.

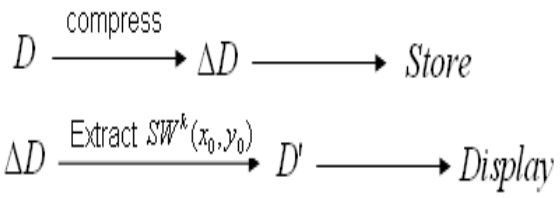

Fig. 3 The DCR problem.

Certainly, these are some factors that should be concerned in the DCR problem such as *the compressed time*, *compressed ratio* and the *extracted time*. Although the compressed ratio is traditionally considers as the most important factor among all, however these two left ones are regarded as the crucial consideration when running the system on Web environment for online processing. In fact, the smaller the ratio is, more time is added for compression and extraction. Hence, we should put a priority for these processing times in the proposed algorithm.

3. Related Researches

Boucheron et al [1] [2] presented the first idea of DEM Compression for fast and efficient search. In these literatures, the authors used some Wavelet transforms such as 5/3, 9/7, max and min wavelets to decompose the original terrain into sub-bands, such that lower sub-bands correspond to higher terrain frequencies and higher sub-bands correspond to lower terrain frequencies, where most of the image energy is concentrated. And then, coefficients are quantified before encoding. After quantification, every coefficient can be encoded by SPIHT coding [21] which transmits the most important image information first. The searching process is fast, however, the compression one is reversed due to a lot of computation is required in SPIHT and Wavelet transforms. The compressed ratio of this method is better than JPEG [18]. Nevertheless, the retrieval operation is not similar to the one in DCR and this algorithm is lossy. Indeed, no further comparison is made as well as utilizing the algorithm for our problem is ignored.

David and Derek [3] showed an algorithm for the lossless compression of DEM based on the statistical correlation of terrain data in local neighborhoods. Elevation data are pre-processed by simple linear prediction algorithms such as 3-Point, 8-Point, 12-Point or 24-Point Lagrange. Then, the differences between the predictions and the real elevations are compressed by Arithmetic Coding. This algorithm is simple and obtains fast compressed time. Moreover, the compressed files are less than half the size of encoded DEM by GZIP. However, the algorithm does not support for retrieval process.

ODETCOM [20] is another method designed for DEM compression. It is a predictor model using a causal template of size eight (linear predictor). To find a compromise for the best set of the coefficients *x*, the authors use an over-determined system of linear equations where each equation corresponds to a prediction and consists of the eight elevations in the causal template and a constant term. However, the compressed time depends on the time solving over-determined linear equation systems. Therefore, it takes long time in case of large sizes of matrix. The compressed ratio is better than JPEG 2000 [18] (lossless mode) and JPEG-LS [22]. Similar to previous method, ODETCOM does not support for retrieval process.

Finally, *ODETLAP* [27] is the newest DEM compression algorithm. This method approximates the original DEM by a set of points *S* (representatives). Then, the (x, y) coordinates are compressed using an adaptive run length encoding method. The *z* sequence is compressed using linear prediction and then by *bzip2*. Indeed, the compressed time is fast. However, this method is lossy and no retrieval process is supported.

Follow our requirements in the Section 2, the proposed algorithm should give fast compressed and extracted times. Therefore, an amelioration of David and Derek method [3] for the DCR problem is the most suitable solution in this case.

4. DCR Scheme

4.1 General Description

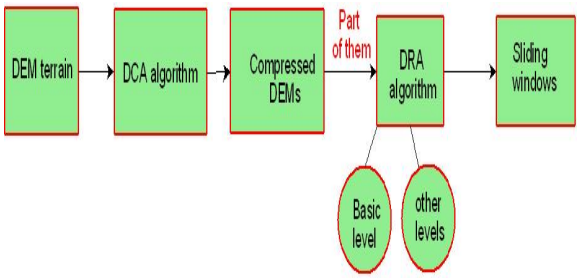

Fig. 4 The DCR scheme

The DCR Scheme acts through the following processes (Fig. 4)

- Client selects a specific DEM terrain in local machine.
- This terrain is compressed by the DCA algorithm and divided into some small compressed sliding windows.
- These compressed data are transferred to Server.
- When Client requests a sliding window, this task is put into the DRA algorithm.
- Depending on which level that being viewed by Client (basic/ others), the DRA will select an appropriate extracted method and return that sliding window.
- The process is repeatedly performed for other requests of sliding windows in any level of moving steps.

4.2 The DCA algorithm

This algorithm runs in Client's side and is dedicated to the compression of a DEM terrain with sizes (W_0, H_0) (Fig. 5, Fig. 6).

Step 1: Find $M = \max(W_0, H_0)$

Step 2: Calculate the smallest n that satisfies

$$2^n \geq M \quad \text{for} \quad n \in N \qquad (7)$$

Then, sizes of the standard DEM in equivalent to the inputted DEM terrain are $2^n \times 2^n$.

Step 3: Specify the acceptance range of moving steps $[K_0, K_{\max}]$ *from the inputted DEM terrain.*

Step 4: Choose some specific levels in this range. For example

$$K_1 = K_0 + 1; \quad K_2 = K_0 + 3; \quad (K_{\max} \geq K_0 + 3)$$

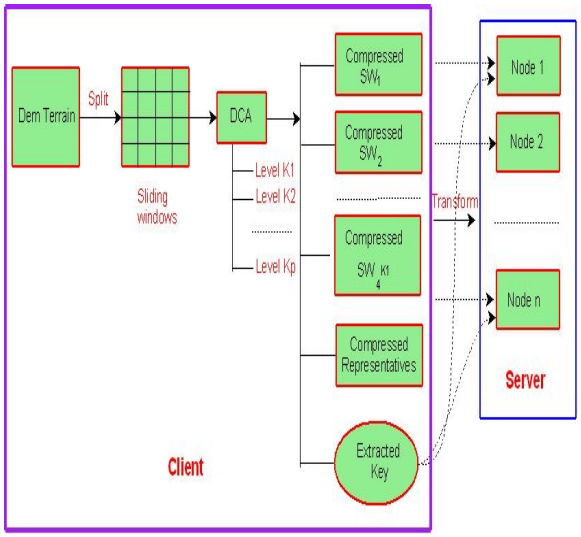

Fig. 5 The DCA algorithm

15	20	21	22	23	26	30
18	17	19	21	32	33	35
10	15	14	13	19	21	25
19	8	7	12	13	14	17
22	23	28	40	35	32	26

Fig. 6 A DEM terrain with $(W_0, H_0) = (7,5)$.

Step 5: Divide the inputted DEM terrain into some sliding windows following by the smallest level K_1 and the sizes of the standard DEM.

a. $T = K_{\max}$

b. From current blocks, start dividing them for level $K_{\max} - T + 1$

b1. If the width of current block is larger than 2^{T-1} then divide it into 2 parts: $[1, 2^{T-1}]$ and $[2^{T-1}, W_0]$

b2. If the height of current block is larger than 2^{T-1} then divide it into 2 parts: $[1, 2^{T-1}]$ and $[2^{T-1}, H_0]$

c. $T = T - 1$

d. Repeat from Step 5b to 5c until $T < K_{\max} - K_1 + 1$

e. Count the total blocks in level K_1 and other levels selected in Step 4. Make a small picture for each level

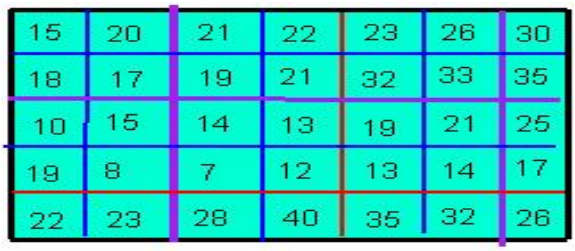

Fig. 7 Sliding windows at level 2.

Step 6: Number all sliding windows at level K_1

```
so = 0
Mark (level k, cell x, cell y)
  If k = 1 then
     If (block (x, y) exists) then ID(x, y) = so++;
     If (block (x, y+1) exists) then ID(x, y+1)= so++;
     If (block (x+1, y) exists) then ID(x+1, y)= so++;
     If (block (x+1, y+1) exists) then ID(x+1, y+1)=so++;
  Else
     Mark (k - 1,x, y)
     Mark (k - 1,x, y + 2^(k-1))
     Mark (k - 1,x + 2^(k-1), y)
     Mark (k - 1,x + 2^(k-1), y + 2^(k-1))
  End if
End Procedure
```

Ex: Mark (2, 1, 1).

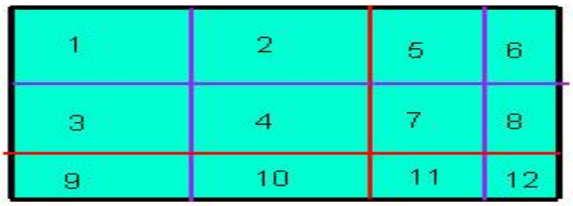

Fig. 8 Numbered sliding windows.

Step 7: Find a representative for each sliding window.

$$R_i = R_{SW}^{K_1}(x_i, y_i) = \{VL(x', y') \mid (x', y') \in SW^{K_1}(x_i, y_i); \left| VL(x', y') - \frac{VL_{\max} + VL_{\min}}{2} \right| \to \min \}$$

$$i \in [1, 4^{K_1}] \quad , (8)$$

Where $VL(x', y')$ is the elevation value at cell (x', y'), $VL_{\max}$ and $VL_{\min}$ are the maximal and minimal elevation values, respectively, in the sliding window $i = SW^{K_1}(x_i, y_i)$, (x_i, y_i) is the original point.

Ex: $R_1 = 18$, $R_2 = 21$.

Step 8: Find median R_e *in the sequence* $\{R_i \mid i \in [1, 4^{K_1}]\}$

Ex: $R_e = 21$

Step 9: Use the David and Derek method [3] *with 1-point Lagrange for all sliding windows. A minor change applied in this algorithm is the using of the representative of each sliding window as the template point instead of the leftmost, bottommost point. Moreover, the coding algorithm for the Corrections is Arithmetic coding instead of using the compression software GZip in* [3]. *The whole process is paralleled computed following by the number of processors or threads in Client. Outputs of this step are compressed sliding windows and positions of representatives in sliding windows (Fig. 9).*

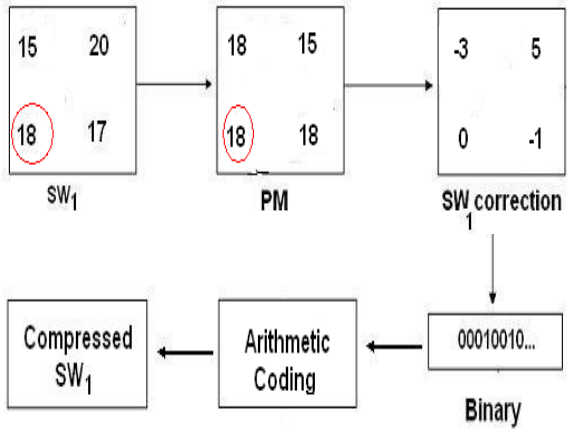

Fig. 9 Modified David and Derek method.

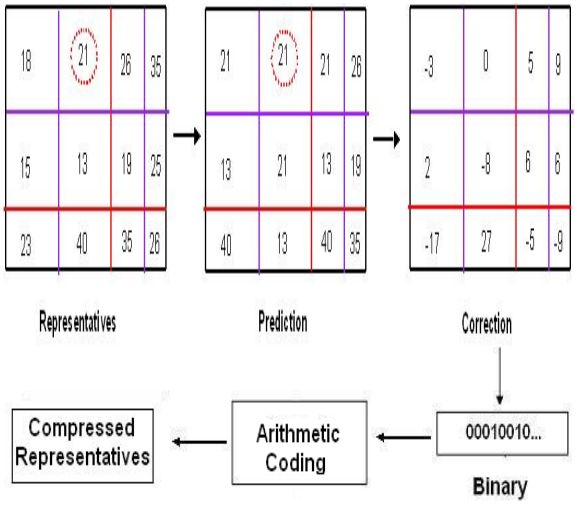

Fig. 10 Representative Compression.

Step 10: Do similar operations for the matrix of representatives with the template point R_e *(Fig. 10) and two other matrices of positions of representatives in sliding windows. We will get the Additional Information including a compressed representative, an extracted key* R_e*, positions of* R_e *in the matrix of representatives (*Pos_R_e*), compressed positions, and two extracted positions (*Pos_x , Pos_y*).*

Step 11: Transfer these data to Server. Depending on the number of processors in the system (np), Server will split these data into all nodes. Except Master node containing the Additional Information, the other nodes store some compressed sliding windows described in the formulae below.

$$No\_p = \frac{K_1^{\,2}}{np} \quad , (9)$$

$$\mathrm{Re}\,mains = K_1^{\,2} \% np$$

Where No_p is the number of compressed sliding windows in each node. $\mathrm{Re}\,mains$ is the number of extra compressed sliding windows that some first nodes have to undertake.

The DCA algorithm stops when the compressed data are totally transferred to Server.

4.3 The DRA algorithm

The DRA algorithm is used to extract some compressed sliding windows following by the selected levels in Step 4 of DCA algorithm. Hence, these are two different extracted methods in equivalent to the level (basic/ others).

For requests to the basic level K_1, some steps are carried out as follow (Fig. 11).

1. *Client sends a request of the sliding window* SW_3 *to Sever.*
2. *Server will transfer the Additional Information in Master node and the compressed sliding window* SW_3 *to Client.*
3. *Client uses the Additional Information to generate* R_3.
4. *From the compressed sliding window* SW_3 *and* R_3 *, Client outputs the result.*

Similar process is applied for requests to others sliding windows. However, Server is not required to transfer the Additional Information again because they were sent to Client in the previous request. This definitely reduces the processing time.

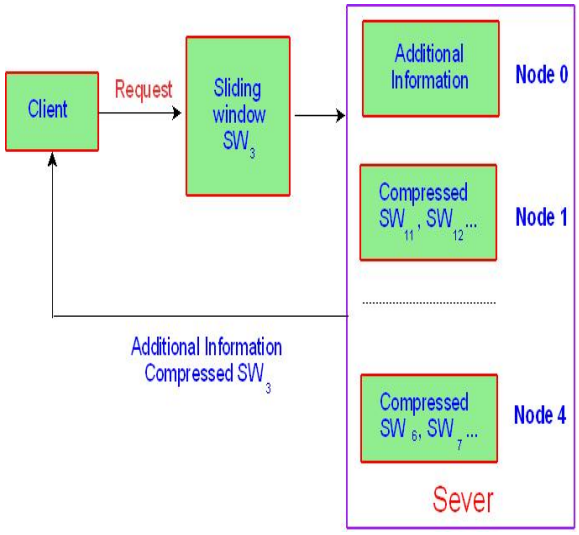

Fig. 11 The DRA algorithm for basic level K_1

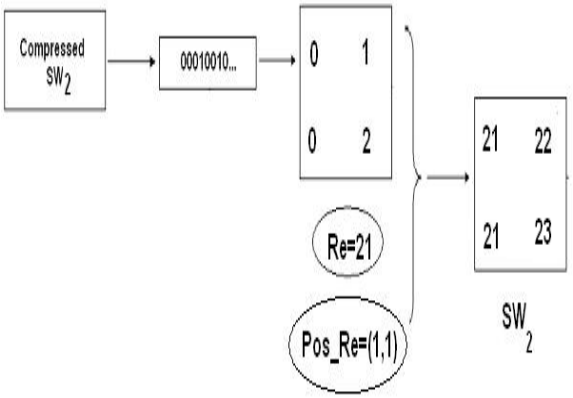

Fig. 12 Extraction Sample.

For other requests to sub-levels, we perform the procedure (Fig. 13).

1. Client sends a request of a sliding window in the sub-level, for example SW_9 in K_2 to the server.
2. Client locates the positions of parent of SW_9 in basic level K_1.

```
Find_Parent (row, col, noRow, noCol, Size, subSize)
Begin
   for i = 0 to noRow - 1
      if (i * size < row * subSize && row * subSize
                                        <= (i + 1) * size)
         row0 = i + 1
         break
      end if
   end for
   for j = 0 to noCol - 1
```

```
        if (i * size < col * subSize && col * subSize
                                        <=(i + 1) * size)
            col0 = j + 1
            break
        end if
    end for
End
```

Ex: In Fig. 14, parent of SW_7 with (row, col) = (2, 3), (noRow, noCol) = (3, 4), and (Size, subSize) = (512, 256) in level K_3 is SW_2 in level K_2.

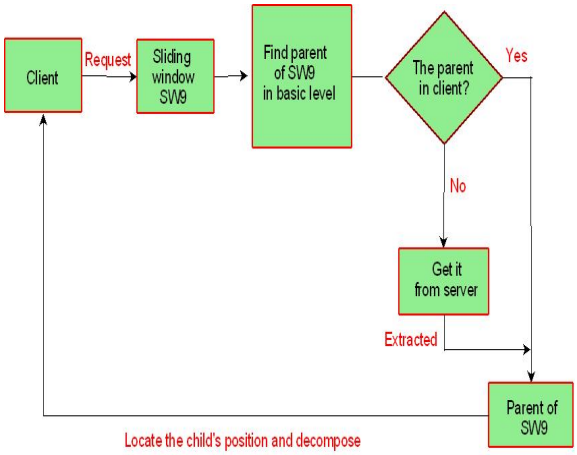

Fig. 13 The DRA algorithm for other levels.

3. Find the ID of parent in level K_1.

FindID (level k, cell x, cell y, ID)

If k = 1 then

If $x \in [1,2^{k-1}]$ and $y \in [1,2^{k-1}]$ then return ID

If $x \in [1,2^{k-1}]$ and $y \in [2^{k-1}+1,2^{k}]$ then return ID+1

If $x \in [2^{k-1}+1,2^{k}]$ and $y \in [1,2^{k-1}]$ then return ID+2

If $x \in [2^{k-1}+1,2^{k}]$ and $y \in [2^{k-1}+1,2^{k}]$ then return ID+3

Else

If $x \in [1,2^{k-1}]$ and $y \in [1,2^{k-1}]$ then

FindID (k - 1, x, y, ID)

If $x \in [1,2^{k-1}]$ and $y \in [2^{k-1}+1,2^{k}]$ then

FindID (k - 1, x, $y-2^{k-1}$, $4^{k-1}+ID$)

If $x \in [2^{k-1}+1,2^{k}]$ and $y \in [1,2^{k-1}]$ then

FindID (k - 1, $x-2^{k-1}$, y, $2*4^{k-1}+ID$)

If $x \in [2^{k-1}+1,2^{k}]$ and $y \in [2^{k-1}+1,2^{k}]$ then

FindID(k-1, $x-2^{k-1}$, $y-2^{k-1}$, $3*4^{k-1}+ID$)

End if

End

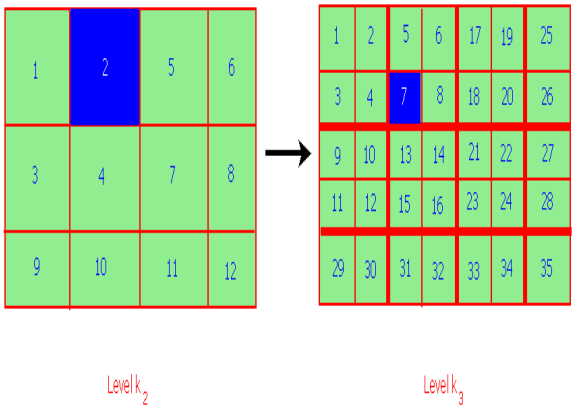

Fig. 14 Find parent.

4. If the parent of SW_9 is not in Client, get it from Server and the Additional Information if missing.
5. Locate the child's positions from the parent and decompose it. Notice that four children of (i, j) at level *k* to level *k +1* are

$$(2i-1,2j-1);(2i-1,2j)$$
$$(2i,2j-1);(2i,2j) \quad . (10)$$

Ex: See Fig. 15.
The final result is the child being displayed in the system.

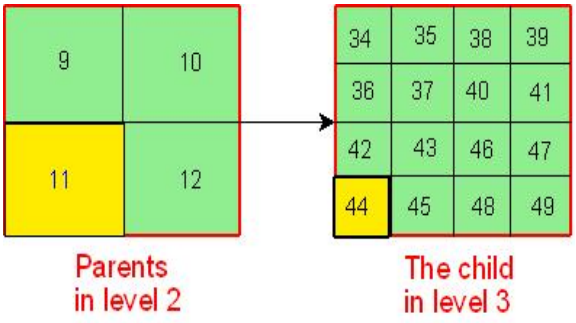

Fig. 15 An example of finding child's position.

5. Evaluation and Simulation Results

Comparing with David and Derek method [3], our method obtains the following characteristics.

- ***Better compressed time***: the total compressed time of DCA includes the dividing, marking, finding medians, and the transform times for all

sliding windows, a representative matrix, and two positions matrices. Hence, it is calculated as

$$T = 2W_0H_0 + 4^{K1} + \frac{4^{K1}}{l}\left[\left(\frac{W_0}{2^{K1}}-1\right)\frac{H_0}{2^{K1}} + \left(\frac{H_0}{2^{K1}}-1\right)\right] + \frac{3}{l}\left[\left(2^{K1}-1\right)2^{K1} + \left(2^{K1}-1\right)\right] \approx \left(2+\frac{1}{l}\right)O(N^2) \le T(David\ \&\ Derek), \quad (11)$$

Where l is the number of processors at Client.

- ***Similar compressed ratio***: the David and Derek method relies on the statistical correlation between points (elevations). Indeed, it is the best choice for compressed ratio. However, the only limitation of this method is that we must use the whole DEM correction for extraction although a part of it is required. Thus, our method generalizes the above method by focusing on the statistical correlation between regions or sliding windows. Because there exists at least a region whose form is different from each ones. Beside, the chosen point in each sliding window is its median instead of the leftmost, bottommost point (Fig. 16). Therefore, the correction matrix certainly contains smaller values than the ones in a matrix using the old method. However, we still have to calculate three more matrices that are representatives and positions. Consequently, the compressed ratio of our method with David and Derek's is the same.

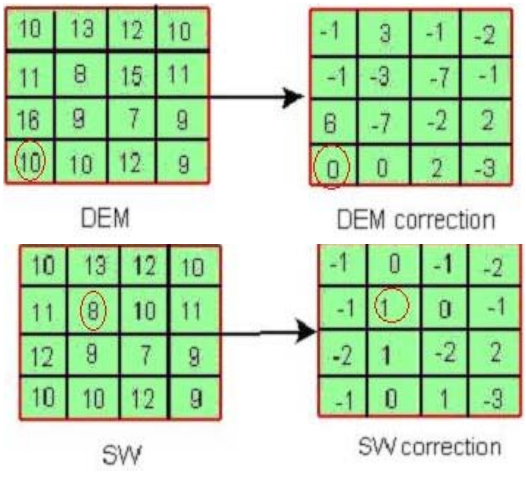

Fig. 16 The different of two methods.

- ***Fast display time of sliding windows***: in the DRA algorithm, only the Additional Information is downloaded and extracted once time. Since then, other sliding windows have utilized it for extraction without downloading again. Thus, the display time is getting faster. In general, it is equal to $1/4^{K1}$ of the extracted time of David and Derek method. Moreover, most of works are processed in the Client's side. Therefore, it helps accelerating the display time as well as prevents Server from falling into overload.

To re-confirm our consideration, we perform some simulations. Indeed, we have implemented the DCR method in Java programming language and executed it on a PC Intel Pentium 4, CPU 2.66GHz, 1GB RAM, 80 GB HDD. Tested data are a benchmark set of 24 USGS 1:250,000 scale DEMs on a 1201 x 1201 grid (see: http://dds.cr.usgs.gov/pub/data/DEM/250/). Each DEM is the first in each higher level USGS directory (ordered alphabetically) for which there is a DEM entry (there are no DEMs for X or Y). Summary statistics of these data can be found at Ref [3], including the elevation ranges, mean elevations, standard deviations, and the entropies or information contents of the original data.

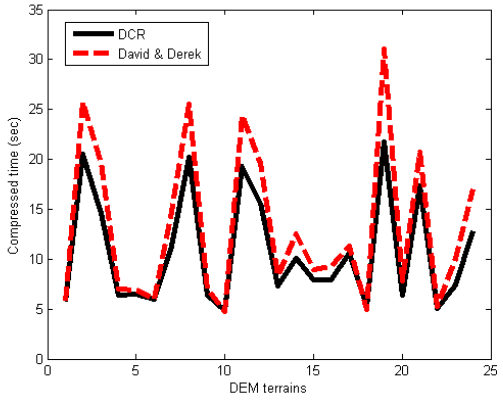

Fig. 17 Compressed times of two algorithms.

The following figures (Fig. 17, Fig. 18) show the compressed time, and compressed ratio of our method in comparison with the David and Derek method [3]. From these results, we can recognize that DCR obtains better compressed time than David and Derek, about 70% in average. Additionally, the compressed ratio of DCR approximates to the one of David and Derek. In some cases, our ratios are much better.

Fig. 19 shows the average display times of sliding windows of all DEM terrains with level $K_1 = 3$. Obviously, these times are really small, about 11% in average of the DCR's compressed times (Fig. 17). The higher the level is, the smaller the display times become.

Therefore, we can perform some analysis actions on a specific sliding window in short time.

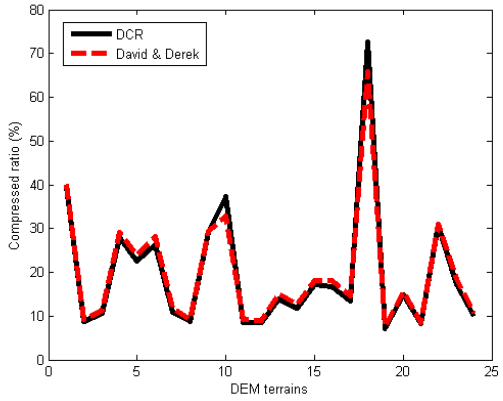

Fig. 18 Compressed ratios of two algorithms.

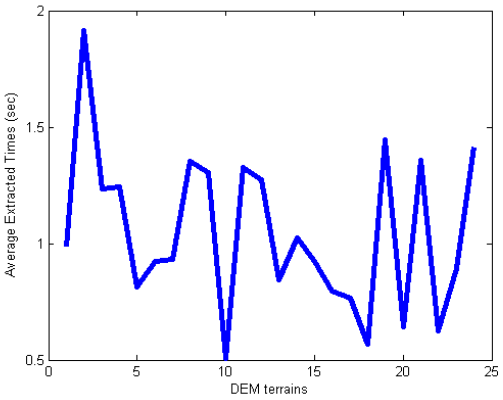

Fig. 19 Display times of DCR method.

6. Conclusion

In this paper, we have presented a modified version of the David and Derek method [3] for the DCR problem. The proposed method is based on the concept of sliding windows and parallel computation and contains two phases: the compression DCA and the extraction DRA. Both processes obtain fast processing time, that is, a good condition to deploy the method through Web environments. The evaluation results in Section 5 have shown that our method is better than the David and Derek method and is totally proved the suitability for the DCR problem.

In the future, we will consider further retrieval operations such as elevation searching or attribute's information query in compressed conditions. In addition, some applications of our method in real situations are also our work.

Acknowledgment

The authors would like to express the cordial thanks to Prof. Pham Ky Anh, Prof. Nguyen Dinh Hoa, VNU, Prof. Pier Luca Lanzi, Dr. Roberto Collonelo, Politechnico di Milano and the research group at Center for High Performance Computing, VNU for their valuable advices. This work is supported by a research grant of Vietnam National University, Hanoi for promoting Science and Technology.

References

[1] Boucheron, L.E., Creusere, C.D, "Compression of digital elevation maps for fast and efficient search and retrieval", Proceedings of International Conference on Image Processing (ICIP 2003), 14-17 Sept 2003, Barcelona, Spain, vol 1, pp. 629-632.

[2] Boucheron, L.E., Creusere, C.D, "Lossless wavelet-based compression of digital elevation maps for fast and efficient search and retrieval", IEEE Transactions on Geoscience and Remote Sensing, Vol 43, Issue 5, 2005, pp. 1210 – 1214.

[3] David B. Kidner, Derek H. Smith, "Advances in the data compression of digital elevation models", Computers & Geosciences, Vol 29, 2003, pp. 985–1002.

[4] David Salomon, "Data compression, the complete reference", Fourth Edition. Springer-Verlag London Limited, ISBN 1-84628-602-6, Section 5, pp. 553-674, 2007.

[5] GAO Ying-jie et al, "Development of DEM on 1:10000 Scale and Its Application in Geo-sciences", Journal of Anhui Agricultural Sciences, 2009-02.

[6] Howard Zhou, Jie Sun, Greg Turk, James M. Rehg, "Terrain Synthesis from Digital Elevation Models", IEEE Transactions on Visualization and Computer Graphics, Vol 13, No 4, July/Aug. 2007, pp. 834-848, doi:10.1109/TVCG.2007.1027.

[7] James A. Thompson, Jay C. Bell, Charles A. Butler, "Digital elevation model resolution: effects on terrain attribute calculation and quantitative soil-landscape modeling", Geoderma 100, 2001, pp. 67–89.

[8] Jan Seibert, Brian L. McGlynn, "A new triangular multiple flow direction algorithm for computing upslope areas from gridded digital elevation models", Water resources research, Vol 43, 2007, W04501, 8 PP, doi:10.1029/2006WR005128.

[9] Kevin J. McMaster, "Effects of digital elevation model resolution on derived stream network positions", Water resources research, Vol 38, 2002, pp. 1042-1050, doi:10.1029/2000WR000150.

[10] LIJuan ZHAO Jun, "Slope-Extraction Analysis Of Northwest Droughty Area Based On DEM", Beijing Surveying and Mapping, 2008-01.

[11] LIU Yongqiong, WU Yanlan, HU Hai, HU Peng, "Assessment method of digital elevation models accuracy and its limitations", Journal of Geomatics, 2009-05.

[12] Li, Z., Zhu, Q. and Gold, C, "Digital terrain modeling: principles and methodology", CRC Press, Boca Raton, 2005.

[13] Le Hoang Son, "On the Development of Three Dimensional WebGIS Systems: Some New Trends and Prospects", Proceedings of the 2010 3rd IEEE International Conference on Computer Science and Information Technology (IEEE

ICCSIT 2010), July 9 - 11, 2010, Chengdu, China, Vol 1, pp. 182 - 186, ISBN 978-1-4244-5537-9, DOI = http://dx.doi.org/10.1109/ICCSIT.2010.5564074.
[14] Le Hoang Son, "An Approach to Construct SGIS-3D: a Three Dimensional WebGIS System Based on DEM, GeoVRML and Spatial Analysis operations", Proceedings of the 2nd IADIS International Conference Web Virtual Reality and Three-Dimensional Worlds 2010 (IADIS Web3DW 2010), July 27 - 29, 2010, Freiburg, Germany, pp. 317 – 326.
[15] Le Hoang Son, "An Exploratory Study about Spatial Analysis Techniques in Three Dimensional Maps for SGIS-3D systems", Proceedings of the 2010 IEEE International Conference on Electronics and Information Engineering (IEEE ICEIE 2010), August 1 - 3, 2010, Kyoto, Japan, Vol 1, pp. 199 - 203, ISBN: 978-1-4244-7680-0, DOI = http://dx.doi.org/10.1109/ICEIE.2010.5559893
[16] Le Hoang Son, Pham Huy Thong, Nguyen Duy Linh, Truong Chi Cuong and Nguyen Dinh Hoa, "Developing JSG Framework and Applications in COMGIS Project", International Journal of Computer Information Systems and Industrial Management Applications (IJCISIM), Vol 3, 2011, pp. 108-118.
[17] M. van Kreveld, "Digital Elevation Models: overview and selected TIN algorithms", In M. van Kreveld, J Nievergelt, T. Roos and P. Widmayer, editors, Algorithmic Foundation of GIS, Springer-Velag, 1997.
[18] Marcellin, M.W.; Gormish, M.J.; Bilgin, A., Boliek, M.P, "An overview of JPEG-2000", Proceedings of Data Compression Conference (DCC 2000), 2000, pp. 523 – 541.
[19] MA Chi, SONG Wei-dong, "Creating urban DEM by use of topographic map with DWG format", Journal of Anshan University of Science and Technology, 2004-05.
[20] Metin Inanc, "Compressing terrain elevation datasets", Dissertation, Rensselaer Polytechnic Institute, Troy, New York, 2008.
[21] Said and W. A. Pearlman, "A new fast and efficient image codec based on set partitioning in hierarchical trees", IEEE Symposium on Circuits and Systems, Chicago, May 1993.
[22] Shantanu D. Rane and Guillermo Sapiro, "Evaluation of JPEG-LS, the New Lossless and Controlled-Lossy Still Image Compression Standard, for Compression of High-Resolution Elevation Data", IEEE Transactions on Geoscience and remote sensing, Vol 39, No 10, 2001, pp. 2298 – 2306.
[23] Sandra Lanig, Arne Schilling, Beate Stollberg and Alexander Zipf, "Towards Standards-Based Processing of Digital Elevation Models for Grid Computing through Web Processing Service (WPS)", Computational Science and Its Applications (ICCSA 2008), Lecture Notes in Computer Science, 2008, Volume 5073/2008, pp. 191-203, DOI: 10.1007/978-3-540-69848-7_17.
[24] SHEN Zhong-yuan, LI Zhan-bin, LI Peng,WU Jin-hui, "Research on the Algorithms of the Digital Channel Network Extraction Based on Digital Elevation Model", Journal of Water Resources and Water Engineering, 2009-01.
[25] S. Rayburga, M. Thomsa and M. Neave, "A comparison of digital elevation models generated from different data sources", Geomorphology, Vol 106, Issues 3-4, 15 May 2009, pp. 261-270.
[26] WANG Ke-ke, ZHANG Li-chao, PAN Zhen, WANG Qing-shan, ZHANG Shi-quan, "Research on Multiresolution Dynamic Creating Networks Algorithm of DEM based on DirectX", Beijing Surveying and Mapping, 2008-02.
[27] Zhongyi Xie, W. Randolph Franklin, Daniel M. Tracy, "Slope Preserving Lossy Terrain Compression", The SIGSPATIAL Special, Vol. 2, Num. 3, 2010, pp. 19-24.

Le Hoang Son is a researcher at the Center for High Performance Computing, Hanoi University of Science, VNU. He is a member of IACSIT and also member of the editorial board of the International Journal of Engineering and Technology (IJET). His major field includes Data Mining, Geographic Information Systems and Parallel Computing.

Nguyen Duy Linh is a researcher and Master student at the Center for High Performance Computing, Hanoi University of Science, VNU. His research interests include Geographic Information Systems and Grid Computing.

Tran Van Huong is a collaborator at the Center for High Performance Computing, Hanoi University of Science, VNU. His research interests include Geographic Information Systems and Data Compression.

Nguyen Huu Dien is an associate professor at the Center for High Performance Computing, Hanoi University of Science, VNU. His research areas include Fixed point theorems, Optimization methods, Sequential and Parallel algorithms, Analysis and Design information systems, LaTeX.. He is author of many books and book chapters on Applied Mathematics since 1999.