Contents lists available at ScienceDirect

Journal of Hydrology

journal homepage: www.elsevier.com/locate/jhydrol

Research papers

Drainage network extraction from a high-resolution DEM using parallel programming in the .NET Framework

Chao Du[a], Aizhong Ye[a,*], Yanjun Gan[b], Jinjun You[c], Qinyun Duan[a], Feng Ma[a], Jingwen Hou[a]

[a] State Key Laboratory of Earth Surface Processes and Resource Ecology, Faculty of Geographical Science, Beijing Normal University, Beijing 100875, China
[b] State Key Laboratory of Severe Weather, Chinese Academy of Meteorological Sciences, Beijing 100081, China
[c] State Key Laboratory of Simulation and Regulation of Water Cycle in River Basin, China Institute of Water Resources and Hydropower Research, Beijing 100038, China

ARTICLE INFO

Article history:
Received 31 July 2017
Received in revised form 15 October 2017
Accepted 17 October 2017
Available online 26 October 2017
This manuscript was handled by G. Syme, Editor-in-Chief, with the assistance of Abhijit Mukherjee, Associate Editor

Keywords:
High-resolution
Parallel computing
Flow direction
Drainage networks
.NET Framework

ABSTRACT

High-resolution Digital Elevation Models (DEMs) can be used to extract high-accuracy prerequisite drainage networks. A higher resolution represents a larger number of grids. With an increase in the number of grids, the flow direction determination will require substantial computer resources and computing time. Parallel computing is a feasible method with which to resolve this problem. In this paper, we proposed a parallel programming method within the .NET Framework with a C# Compiler in a Windows environment. The basin is divided into sub-basins, and subsequently the different sub-basins operate on multiple threads concurrently to calculate flow directions. The method was applied to calculate the flow direction of the Yellow River basin from 3 arc-second resolution SRTM DEM. Drainage networks were extracted and compared with HydroSHEDS river network to assess their accuracy. The results demonstrate that this method can calculate the flow direction from high-resolution DEMs efficiently and extract high-precision continuous drainage networks.

1. Introduction

A Digital Elevation Model (DEM) is composed of a set of grids, wherein each grid contains both elevation and coordinate information. The morphology of the terrain surface can be obtained from DEMs, including hydrology information. River networks are fundamental parameters for hydrological simulations. The extraction of drainage networks from DEMs has been utilized and improved upon for decades and is currently widely used in the simulation of hydrological processes (Peucker and Douglas, 1975; O'Callaghan and Mark, 1984; Jenson and Domingue, 1988; Martz and Garbrecht, 1998). The most significant components within the traditional method of river extraction are the pretreatment process and flow direction determination. Many studies have focused on developing methodologies to reduce the occurrence of parallel rivers in addition to other problems caused by depression-filling pretreatment (Garbrecht and Martz, 1997; Grimaldi et al., 2007; Poggio and Soille, 2012). Some of these studies proposed a method for calculating the flow direction grid by grid using information pertaining to the outlets of the basin and employing a specific algorithm when the method encounters a sink. This method combines the calculations for sink filling and flow direction and ensures river continuity characteristics (Ye et al., 2005; Mao et al., 2014).

An accurate river network can be obtained effectively using 30-m-resolution DEMs (Poggio and Soille, 2011). However, with an increase in the basin area, the resulting enormous quantity of grids reduces the efficiency of computation. For the purposes of enhancing computational performance, low-resolution DEMs are used more commonly, whereupon the advantages of high-resolution DEMs cannot be fully utilized. Parallel programming is an effective option for algorithms to reduce the cost of computation. The parallel method used to extract drainage systems is generally based on the sub-basin unit, and these parallel techniques have been applied in previous research investigations (Apostolopoulos and Georgakakos, 1997; Grübsch and David, 2001; Wang et al., 2004; Cui et al., 2005).

Studies have selectively used workstations equipped with high memory and storage or a server composed of several computers to extract hydrological information from high-resolution DEMs through the parallel method (Gong and Xie, 2009; Tesfa et al., 2011; Bai et al., 2015). The Message-Passing Interface (MPI) and Open Multi-Processing (OpenMP) interface are widely used in parallel programming. Many algorithms have used these two standards to achieve parallel computing (Clematis et al., 1997;

* Corresponding author.
E-mail address: azye@bnu.edu.cn (A. Ye).

https://doi.org/10.1016/j.jhydrol.2017.10.034
0022-1694/© 2017 Elsevier B.V. All rights reserved.

Li et al., 2006, 2011; Do et al., 2011). Graphics processing units (GPUs) are not used solely within Computer Graphics but also for General Purpose Computation, which can be known as a GPGPU (General Purpose GPU). CUDA (Compute Unified Device Architecture), which was developed by NVIDIA, provides a language for parallel computing that research studies have previously adopted to conduct parallel programming (Ortega and Rueda, 2010; Qin and Zhan, 2012; Rueda et al., 2016; Sten et al., 2016). OpenCL (Open Computing Language), which is similar to CUDA, uses a CPU/GPU to achieve parallel computing (Tristram et al., 2014). A linear solver is another method with which to spread data over multiple computer processors (Richardson et al., 2014). Although there are numerous methods with which to process parallel calculations, relatively few studies have employed the .NET Framework, which provides a series of libraries and classes for parallel computing, and simultaneously simplifies the codes.

This study proposed a parallel method with which to extract a high-resolution drainage network and simultaneously improve the computation efficiency of a high-resolution DEM. The parallel method utilized is parallel programming within the .NET Framework, which uses multiple threads to implement concurrent processing. Section 2 introduces the methodologies, after which Section 3 covers the data and study domain. Section 4 presents the results and corresponding discussion, and Section 5 provides the conclusions.

2. Methodologies

2.1. The framework for the proposed method

The key component for drainage network extraction is the determination of the flow direction. The method of flow direction determination initiates with the outlets, which is different from traditional methods. It traverses a series of arrays, and the time required for each loop is increased with an increase of the number of grids. Compared with the flow accumulation calculation and drainage network extraction steps, the flow direction determination necessitates additional traversals, which means it costs more time. Therefore, we applied parallel computing for the flow direction determination.

Fig. 1 shows a flow diagram for the methods which have been utilized in this study. First, a low-resolution DEM was used to acquire the sub-basins using multiple thresholds. Second, a high-resolution DEM was divided into sub-DEMs based on the sub-basin units, which were regarded as independent basins to determine the flow direction. Because each sub-basin only had one outlet, each sub-DEM also only possessed a single outlet. Third, we used parallel programming in the .NET Framework to calculate the flow direction of each sub-DEM, the results of which were merged into one file that contains the flow direction of the entire basin. Lastly, we extracted the drainage network from the high-resolution flow direction.

2.2. Flow direction determination

The flow direction determination of the grids is generalized by using a 3×3 window scan of the DEM. The calculation methods employed to determine the flow direction are typically divided into the single-flow-direction method and multi-flow-direction method. The former method assumes that the central grid of water flows only in one direction, while the latter assigns the central grid flow to multiple grids, and the water is subsequently assigned to a number of relatively lower-elevation grids according to a proportion. The single-flow-direction method is easier to operate, as it does not require the consideration of many factors. In this paper,

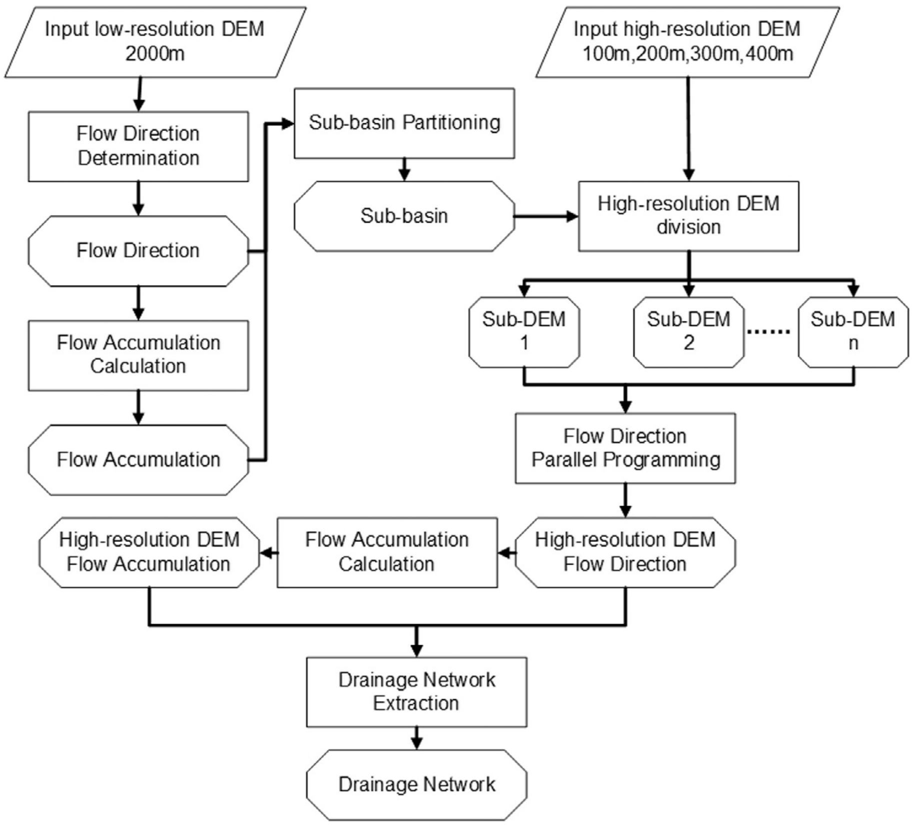
Input low-resolution DEM 2000m
Input high-resolution DEM 100m,200m,300m,400m
Flow Direction Determination
Sub-basin Partitioning
Sub-basin
High-resolution DEM division
Flow Direction
Sub-DEM 1
Sub-DEM 2
Sub-DEM n
Flow Accumulation Calculation
Flow Accumulation
Flow Direction Parallel Programming
High-resolution DEM Flow Accumulation
Flow Accumulation Calculation
High-resolution DEM Flow Direction
Drainage Network Extraction
Drainage Network

Fig. 1. Flow chart of the methods utilized in this study.

a D8 single flow algorithm was adopted to ascertain the flow direction of water leaving the grid according to the degree of the slope. However, in contrast to the traditional approach of scanning each grid individually, the algorithm originated from the outlets of the basin and confirmed the flow direction of each grid gradually. This ensured that each grid's water can flow to the outlets of the basin; however, as a consequence of this approach, the algorithm worked relatively slowly. This method did not require a depression-filling pretreatment, and handled the sinks by forcing water flow into the neighboring grid such that the elevation of the sink grids did not change. This resulted in obtaining the hydrologic information in the depression area, including the drainage network. In addition, it avoided many problems which can be caused by depression-filling, including the presence of parallel rivers.

Fig. 2 illustrates the calculation procession in detail. The grids were divided into two classes: one consisted of determined grids, for which the flow direction was already known, and the other were candidate grids, for which the flow direction would be calculated later. The flow direction of the outlets was confirmed to be 999 and was marked as determined grids. If the grids adjacent to the determined grids did not possess a determined water flow, they would be marked as candidates and sorted by an ascending elevation. Then, the candidates were traversed to conduct the D8 algorithm. If the grid flowed into a determined grid, its flow direction could be determined and removed from the candidates. If a new grid possessed a flow direction, it was added to the candidates along with its neighboring grids that had not been calculated, and all of the grids were sorted by ascending elevation. This process of traversing the candidates was then repeated. If new grids are absent within which to determine the flow direction, it follows that a sink must exist within the basin. Because the neighboring grids are sorted by ascending elevation values, the lowest elevation grid contains the depression. Subsequently, searching the grids adjacent to that with the lowest elevation to examine whether there are determined grids would reveal the grid with the maximum slope relative to the depression grid, which can be selected as the flow grid, although it may not represent the grid with the maximum slope in all of the eight directions. According to the laws of nature, water within a depression cannot flow to other areas. Thus, sink grids may not flow to the lowest elevation grid and thus cannot influence other hydrological information. This method forces the water within a depression to flow to the determined grids, which ensures that water within the sink grids can flow to the outlets.

2.3. Flow accumulation calculation

The flow accumulation was obtained using a method for the simulation of flow on the terrain surface. In this study, we assumed that each grid of the DEM has a unit of water. Considering the flow of water from high to low elevations, flow accumulation can be calculated by the flow direction. The flow accumulation value of the outlets is typically relatively large. If there is only one outlet, the outlet's flow accumulation value is the total amount of the accumulations within the basin grids minus one unit. In this study, the flow accumulation value was computed relative to the catchment area of each grid of the upper reach, wherein the units are in square kilometers.

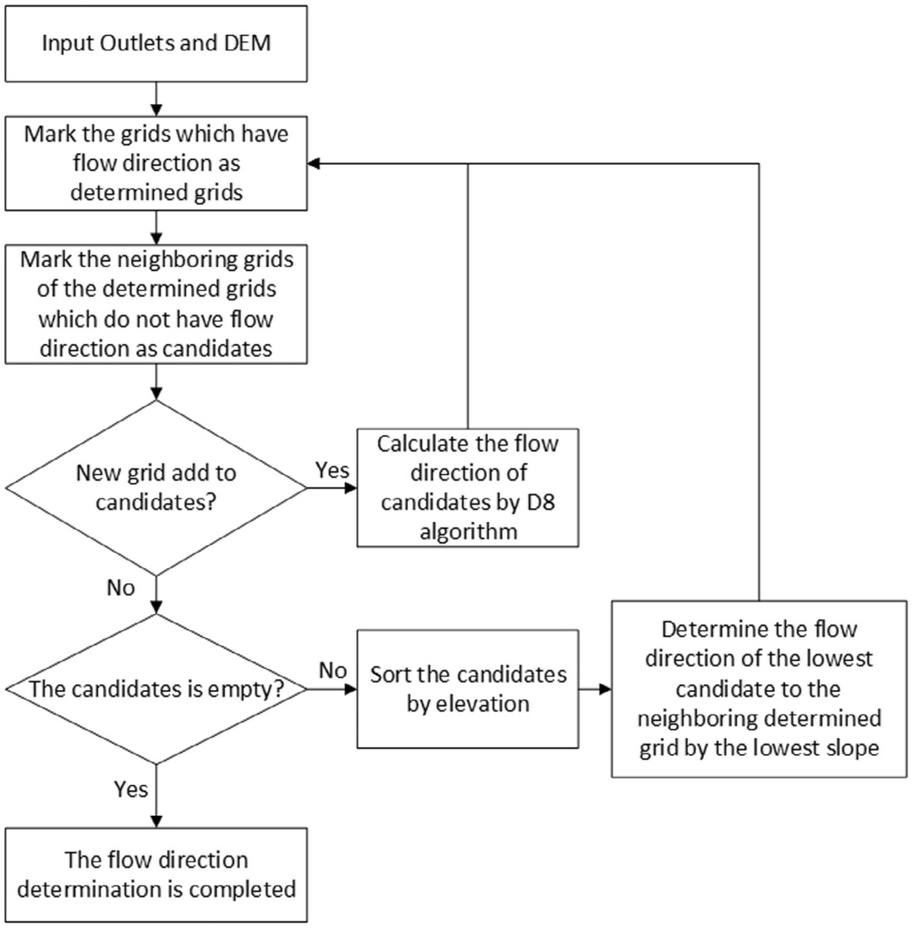
Input Outlets and DEM
Mark the grids which have flow direction as determined grids
Mark the neighboring grids of the determined grids which do not have flow direction as candidates
New grid add to candidates?
Yes
Calculate the flow direction of candidates by D8 algorithm
No
The candidates is empty?
No
Sort the candidates by elevation
Determine the flow direction of the lowest candidate to the neighboring determined grid by the lowest slope
Yes
The flow direction determination is completed

Fig. 2. The steps used in this study to determine the flow direction.

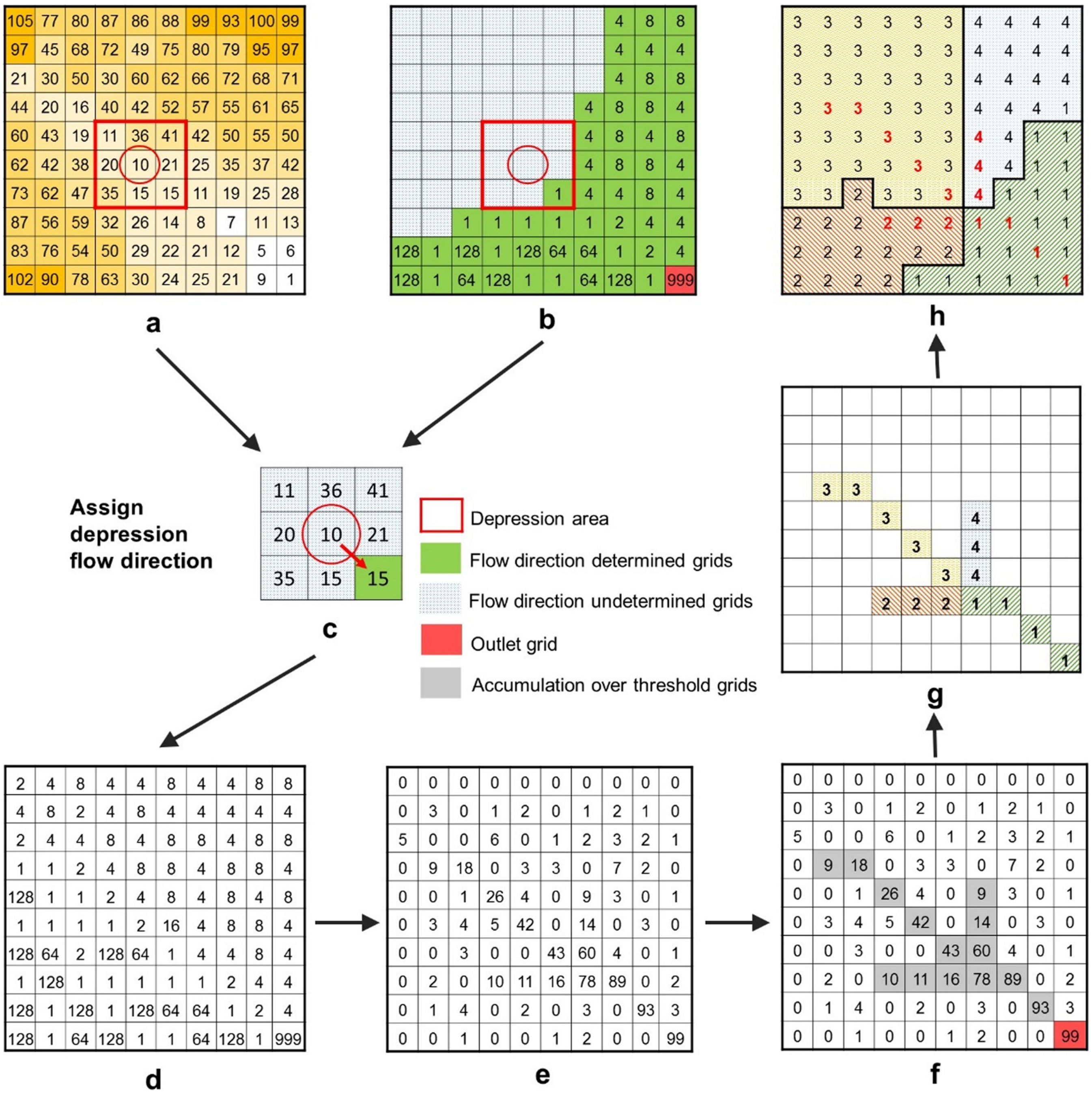
a
b
c
d
e
f
g
h
Assign depression flow direction
Depression area
Flow direction determined grids
Flow direction undetermined grids
Outlet grid
Accumulation over threshold grids

Fig. 3. A detailed example of drainage network extraction and sub-basin partitioning. a) Original DEM, where the number in each grid represents the elevation. b) The calculation for the flow direction determination, wherein the red circle is labeled a sink grid, and the red square indicates that the area is a depression. c) Depression area, wherein the red arrow delineates the flow direction of the sink grid. d) Flow direction of the whole basin. e) Flow accumulation. f) Flow accumulation threshold over 8. g) Drainage network. h) Sub-basins.

2.4. Drainage network extraction and sub-basin partitioning

The preconditions of drainage network extraction are provided by the flow direction and flow accumulation. A threshold was set to limit the flow accumulation, which was called the minimum catchment area of the drainage network. Similar to the flow direction determination, the drainage network extraction began by calculating from the outlets. The outlets also required filtering, following which the accumulation value, which was larger than the threshold, would represent the start of the stream network. The drainage network extraction method contained the following steps. 1) Filter the flow accumulation results with a threshold, and mark the grids containing values larger than the threshold. 2) Initiate the computation from the filtered outlets, and then search the marked grids. If the marked the grid was an inflow grid, it was considered to be a part of the drainage network. 3) If there were two or more inflow girds, the serial number of the river increased to a new number. 4) If there are no more inflow grids, the method signaled the end of the river.

Sub-basin partitioning is based on the drainage network. The calculation was based on the flow direction, flow accumulation and drainage network. Different reaches were numbered following the drainage network extraction. Grids that flowed to one stretch of a river comprised a single sub-basin. As a result, each sub-basin had only one outlet. The method was composed of the following steps. First, search all inflow grids representing the reach of each river based on the serial number from the results of the drainage network. Second, provide all of the inflow grids with the same number as the serial number of the reach. If there existed some small sub-basins whose catchment area was less than the threshold, they were merged with a neighboring sub-basin. Fig. 3 demonstrates an example showing how the flow direction was determined and the steps used to extract the drainage network and sub-basin.

2.5. High-resolution DEM division

The subdivision of the high-resolution DEM is a very significant step in this study. Because high-resolution DEMs contain a large number of grids, they can demand an excessive amount of time and resources to compute. If it can be divided into smaller segments, the calculation process can be simplified. Division based on sub-basins can reduce the connection between different sub-DEMs. Each sub-basin only has one outlet, and the water of this outlet must flow to an adjacent sub-basin on the basis of the drainage network. The flow direction of the outlet is determined by the D8 algorithm. In Fig. 4, sub-basins 2 and 3 flow to sub-basin 1, which means the outlet flow directions of these two sub-basins must point to the grids containing sub-basin 1. The number within each grid is the elevation value, the color represents a different sub-basin, and the outlet is marked in red. In 4c), the black arrow is the flow direction, which points to the grid with an elevation of 360, thereby representing the steepest slope among the grids of sub-basin 1. The outlets of sub-basin 2 and sub-basin 3 flow into sub-basin 1 based on the steepest slope.

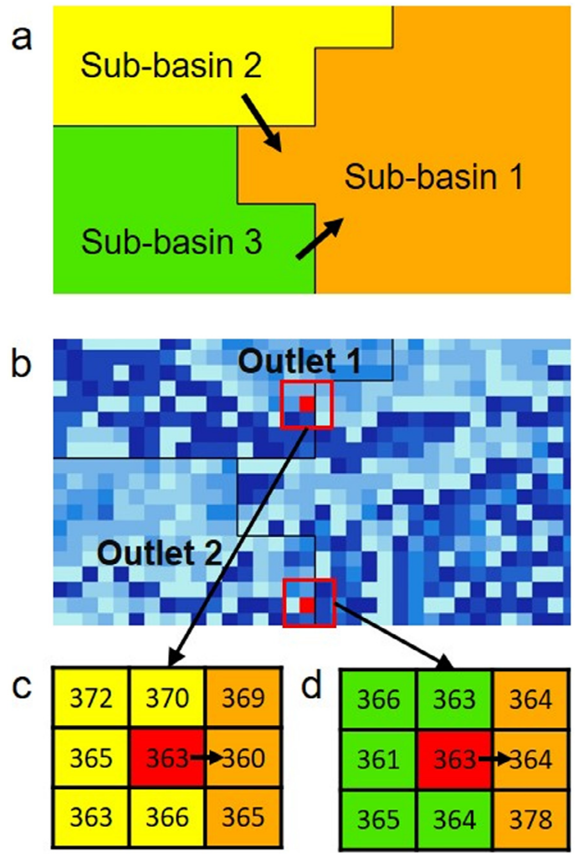

Fig. 4. The determination of the flow direction of each sub-DEM outlet. a) Three sub-basins. b) The outlets of sub-basin 2 and sub-basin 3. Outlet 1 is the outlet of sub-basin 2, while outlet 2 belongs to sub-basin 3. c) Flow direction of outlet 1. d) Flow direction of outlet 2.

2.6. Parallel programming in the .NET Framework

Typically, parallelism is divided into data parallelism and task parallelism. The method employed in this study belongs to data parallelism. Data parallelism refers to simultaneously applying the same operation on elements in a source of collections or arrays. The high-resolution DEM had been divided into sub-DEMs according to different sub-basins. However, because the boundaries of the sub-DEMs were not strictly correlative with different resolutions, the correlation between the sub-DEMs was weak; therefore, the calculations could be conducted separately.

Many personal computers and workstations utilize two out of four cores (CPUs) for a process, which allows for multiple threads to be performed simultaneously. To take advantage of present-day hardware, the code in this work was parallelized to distribute it across multiple processors. Fig. 5 illustrates the method for determining the flow direction of a basin using a parallel computation method. The original DEM is divided into sub-DEMs according to the sub-basins. The flow direction determination of each sub-DEM operates on different threads concurrently, the results of which will be merged into a single entity that represents the flow direction of the entire basin.

The .NET Framework provides a runtime, class library types, and diagnostic tools to support parallel programming. It simplifies parallel code, and does not require working with threads or a thread pool directly. The .NET parallel program framework contains two types: Parallel Queries and Parallel Algorithms. In this study, the Parallel Algorithm type was chosen. Fig. 6 illustrates the architecture of parallel programming in this study. The Task Parallel Library (TPL), which is one of the technologies grouped within the Parallel Algorithms, is a set of APIs and public types in the namespace of System.Threading and System.Threading.Tasks. Multiple threads operate on different segments concurrently through the TPL, which simplifies the process of adding parallelism and concurrency to applications to improve the productivity of developers. It also increases the extent of concurrency dynamically in order for all of the available processors to function efficiently. A TPL is the first choice for the multithreaded code and parallel programming technique within the .NET Framework. The TPL supports data parallelism through the class known as System.Threading.Tasks.Parallel. This class provides the basic usage of For and Foreach loops, which are similar to sequential loops, and consequently developers are not required to create threads or queue items that will operate in the background.

The maximum number of concurrent tasks enabled can be set by the MaxDegreeOfParallelism property. By default, For and ForEach will utilize many of the threads by the program, while MaxDegreeOfParallelism only limits the number of concurrently running operations. If there is no limit set by MaxDegreeOfParallelism, the threads that will be utilized depend on the number of core processors and the load of the machine.

3. Application

3.1. Study area and data

The Yellow River is China's second-largest river. It arises in the northern Qinghai plateau of the Balyanlkalla mountain range, whereupon it winds eastward across the Loess Plateau and the Huang-huai-hai Great Plains, and eventually drains into the Bohai

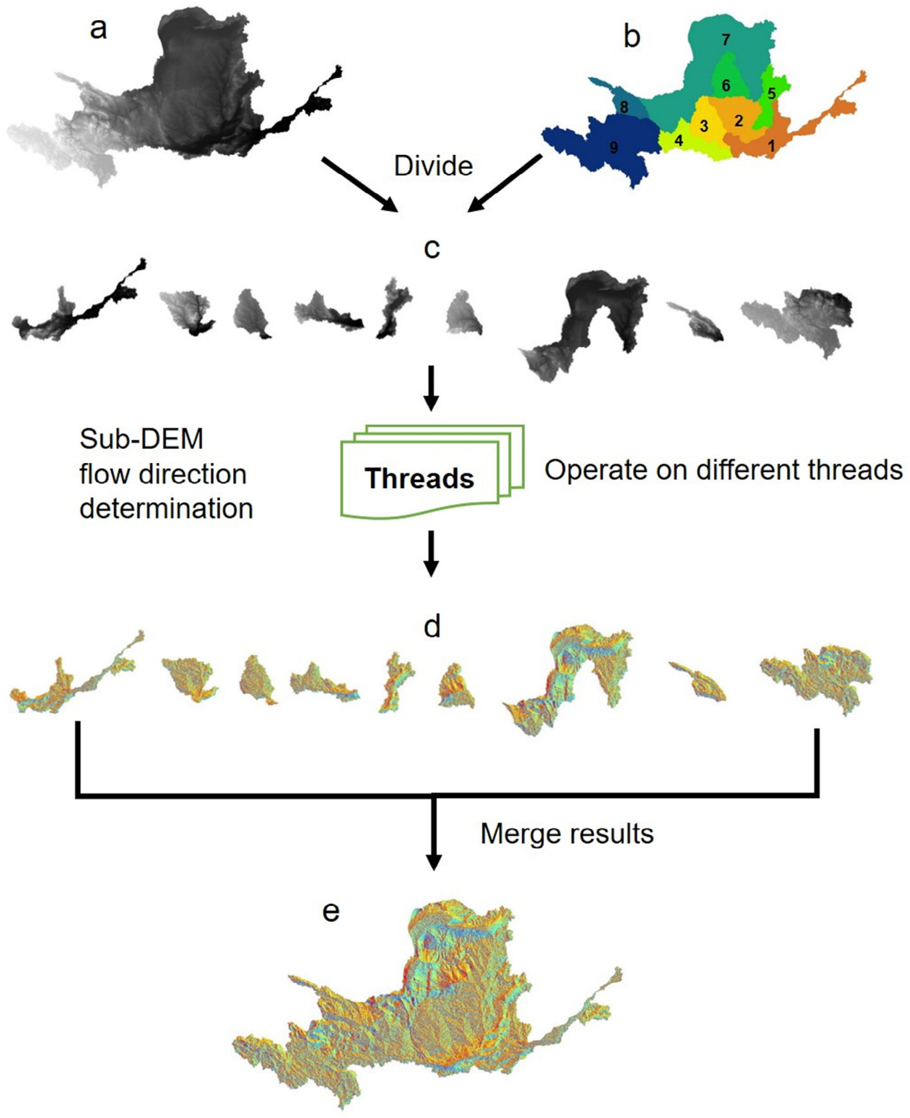

Fig. 5. Parallelism of flow direction determination. a) Original DEM. b) Sub-basin. c) Sub-DEM. d) Flow direction of sub-DEM. e) Flow direction of the basin.

Sea. The length of the main stream is 5464 km, and the water drops across an elevation of 4480 m. The total area of the basin is 795,000 square kilometers (including an interior drainage area of 42,000 square kilometers) (See Fig. 7).

The DEM data was obtained and downloaded from the Shuttle Radar Topography Mission (SRTM) (http://srtm.csi.cgiar.org). The resolution of the original DEM is 3-arc-seconds (90 m). To make the computation processes more efficient, the original DEM was resampled to 100 m, 200 m, 300 m, 400 m and 2000 m. The 2000 m resolution DEM was used as the low-resolution DEM for partitioning the basin, while the others were used as high-resolution DEMs to execute flow direction parallel programming. In Table 1, the file size and grid number exponentially increase with an increase in the resolution. Meanwhile, the min and max elevation have little variation between the different resolutions.

3.2. Computing environment

The code was written in C# language in Microsoft Visual Studio 2013 under the .NET Framework 4.5 platform. The program was executed on one personal computer with an Intel i7-4610 M hyper-threaded quad-core processor, 3.0 GHz CPU frequency, and a memory of 16 GB. The computer system was a Windows 10 64-bit operating system.

4. Results and discussion

4.1. River precision analysis

To evaluate the accuracy of the flow direction, we calculated the river networks and subsequently compared them with river networks from HydroSHEDS (Hydrological data and maps based on SHuttle Elevation Derivatives at multiple Scales), which were obtained from https://hydrosheds.cr.usgs.gov/. HydroSHEDS is a mapping product that provides regional- and global-scale hydrographic information that is derived from 3 arc-second resolution SRTM (Shuttle Radar Topography Mission) data that have been hydrologically conditioned. The river network within HydroSHEDS is selected using the upstream drainage areas exceeding a certain threshold, which here is defined as 100 upstream cells for the 15

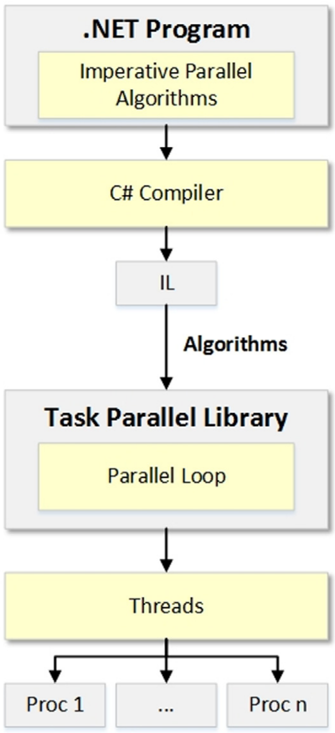

Fig. 6. Parallel programming architecture used in this study.

arc-second resolution data. HydroSHEDS is regarded as high-precision river network data because it is compiled using many methods to reduce error. Fig. 8 demonstrates the corresponding results, wherein the flow directions of the rivers extracted from the four resolution DEMs and the HydroSHEDS river network are alike visually. We transferred the flow accumulation results from the number of cells to the drainage area so that we could set thresholds directly to avoid considering the resolution. We set 100 km^2 as the drainage area threshold to extract river networks.

The qualitative judgment is not rigorous; therefore, we applied a quantitative method to measure the precision. We established a range of buffers for the HydroSHEDS river network, and then performed an overlay analysis of rivers at different resolutions at the buffers. The rivers in the buffer can be considered as the correct rivers, following which we could obtain their length and apply division to obtain the length of the original rivers, and the subsequent result is the accuracy rate. We can see from Table 2 that when the buffer is 1000 m, all of the accuracy rates exceed 90 percent, and the rivers based on the parallel method are more precise. Because the rivers possess a particular width, when the buffer is beneath 100 m, the accuracy rate using a low-resolution DEM is small. Thus, an increase in the buffer size leads to a higher accuracy rate, and high-resolution DEMs performed better than low-resolution DEMs.

4.2. Computing time analysis

4.2.1. Calculation of the flow direction directly

The formula N = A/r^2 can represent the relationship between a DEM resolution and the number of grids, which is a power law function. This suggests that higher-resolution DEMs correspond to a higher number of grids.

If only the D8 algorithm is applied to the grids, the algorithm is required to traverse all of the grids; therefore, the time complexity is O(N), where N is the number of DEM grids. The method we employed in this study begins from the outlet and traverses the candidates array repeatedly, wherein the algorithm sorts all of the grids by elevation gradually and searches their water flow grid by grid simultaneously, which leads to a time consumption that is larger than that of the D8 algorithm. The time complexity of the flow direction determination was O(N^2), and the resulting computation time curve versus the number of grids is shown in Fig. 9. The constant coefficient C only equaled 1.161e-11, but when the grid number is sufficiently large, it will require an excess of computation time. When DEM resolution is 100 m, the grid number is 253151190, which will require a computation time of 206 h according to the fitting formula. General-configuration computers cannot afford these computation times. The memory and storage of such computers are limited, and with an increase in the grid number, they will breach their capacity. Because of this, we could not calculate the flow direction from the 100 m resolution DEM directly.

4.2.2. Calculation of the flow direction using parallel programming

We set different thresholds to partition the sub-basins. The number of sub-basins impacts the parallel computing efficiency. The relationship between the thresholds and sub-basin number is obvious. That is, the sub-basins number is halved if the threshold is doubled.

In this study, we conducted a flow direction determination of sub-DEMs using both a serial and parallel method. The serial method conducts calculations one by one, while the parallel method calculates sub-DEMs simultaneously on a set of threads. Clearly, the parallel method will require less time than the serial method, but the time saved using the parallel method differ according to the threshold, that is, the number of sub-DEMs, which can be observed in Table 3. Four resolution DEMs were used to extract the flow direction of different thresholds using both a serial and parallel method.

The results are plotted in Fig. 10. To find the relationship between the sub-basin number and computational time, the Y-axis is a logarithmic axis. On the whole, the curves from the serial and parallel methods at the same resolution are similar. The curves have little variation between resolutions, and are generally consistent. The time is not strictly increased with an increase of the threshold value. However, when there are too many sub-DEMs, the computing time is increased.

From the perspective of serial and parallel computing, a smaller number of sub-basins number is small leads to a time-consuming computation for the flow direction. The plot is similar to a nike function. Thus, we assumed a nike function relationship between the computing time and sub-basin number. Different resolutions also have a relationship with time, and the grid number is also a parameter.

$$T = a \cdot \frac{S_g}{S_n} + b \cdot \frac{S_n}{S_g} \tag{1}$$

where S_g is the number of grids of the original DEM, S_n represents the number of sub-basins, and T is the computing time. We used a curve-fitting method to solve for a and b. Serial and parallel methods were calculated separately.

Table 4 illustrates the results for a and b under different flow direction computing methods. The R^2 of both methods are over 0.94; therefore, we can draw the conclusion that the fitting effect is reliable.

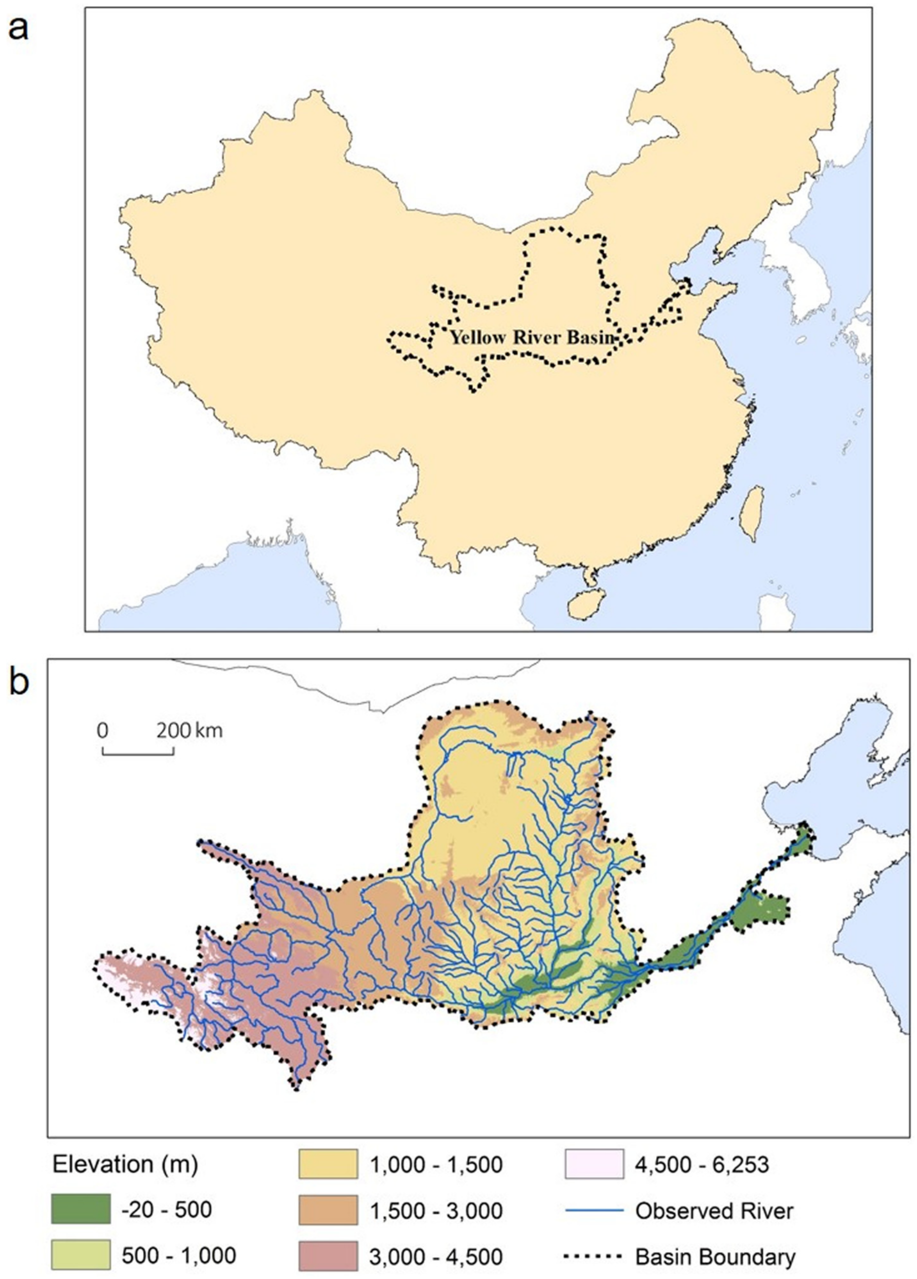

Fig. 7. a) The location of the Yellow River basin. b) 100 m resolution DEM of the Yellow River basin and the observed river network.

Table 1
The properties of different resolution DEMs.

Resolution	File Size (MB)	Min elevation (m)	Max elevation (m)	Columns and rows	Valid grid number
100 m	965.70	−20	6253	21,510 × 11769	80652902
200 m	241.44	−12	6250	10,755 × 5885	20163399
300 m	107.30	−16	6243	7170 × 3923	8961515
400 m	60.36	−13	6252	5378 × 2942	5040784
2000 m	2.41	−4	5982	1076 × 588	201640

Fig. 11 demonstrates two types of concurrent processing. In Fig. 11a, the size of the sub-DEM is similar; therefore, the two threads can be used effectively. However, in Fig. 11b, the sub-DEM is very large; therefore, when the other sub-DEMs are being concluded, it will require more time for calculation. In this case, the advantage of having multiple threads is not obvious, since one thread may necessitate a longer time than the other. In this study, when the threshold reaches 32,000 km^2, there are 9 sub-DEMs, although one sub-basin, delineated as number 7, too large (Fig. 12). This sub-DEM fits into the second type of concurrent

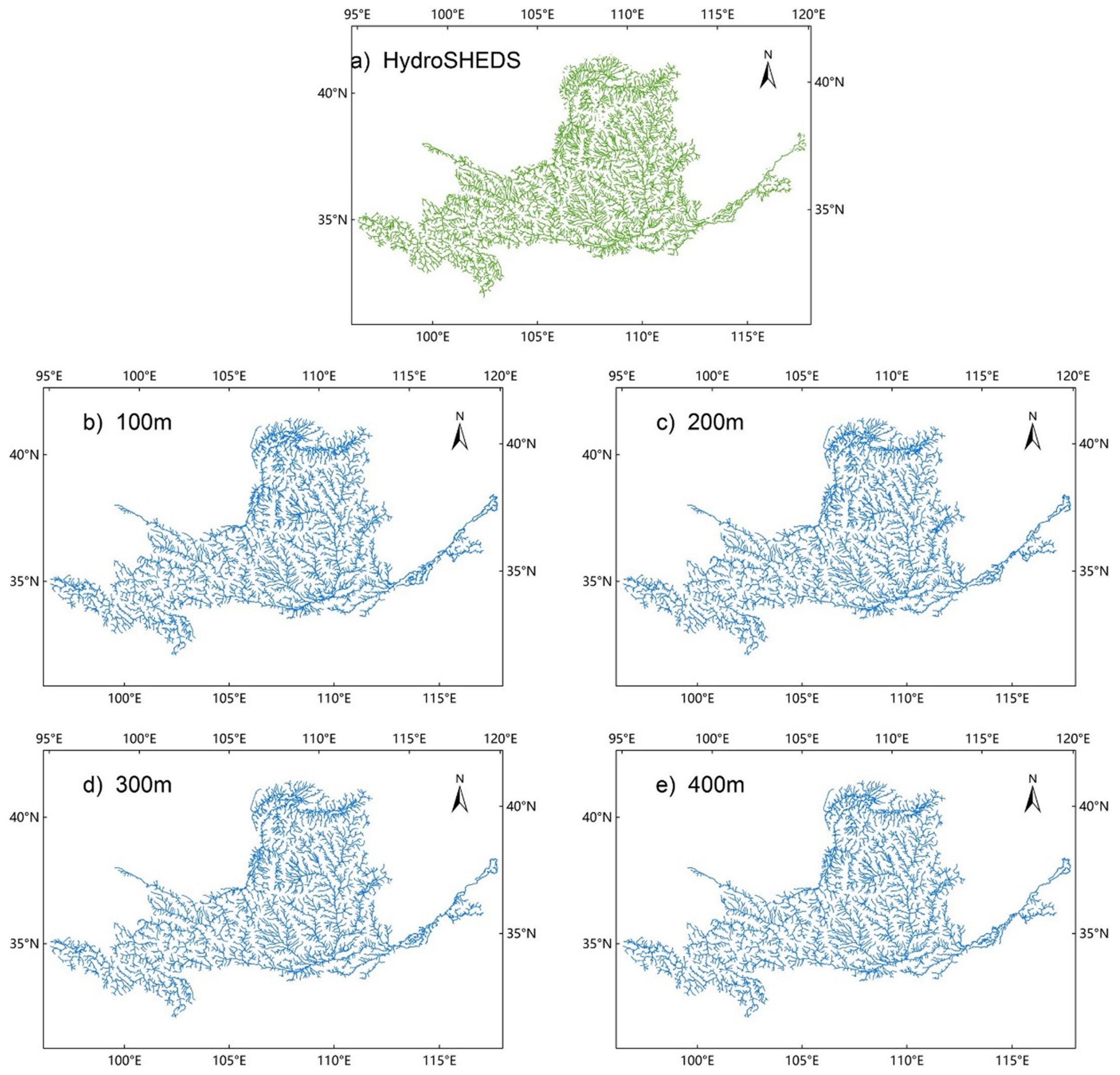

Fig. 8. a) HydroSHEDS river network of the Yellow River. b-e) Drainage networks extracted from 100 m, 200 m, 300 m, 400 m resolution DEMs, respectively.

Table 2
The extraction accuracy rate of river networks for different resolution DEMs at different buffers of HydroSHEDS river networks.

Resolution(m)	Buffer(m)					
	100	200	300	400	500	1000
100	88%	95%	95%	95%	96%	96%
200	62%	76%	80%	83%	85%	90%
300	29%	55%	72%	80%	83%	90%
400	27%	52%	69%	78%	82%	90%

processing, and made that the parallel cost was almost the same as that of the serial.

For the purposes of analyzing parallel efficiency, we calculated the speed ratio, which is shown in Fig. 13. The speed ratio ranges from 1.1 to 2.1. All of the resolution DEMs have a similar speed ratio. They all peak at 77 sub-basins and bottom out at 9 sub-basins. Thus, when the number of sub-basins is 77, the threads could be utilized most effectively.

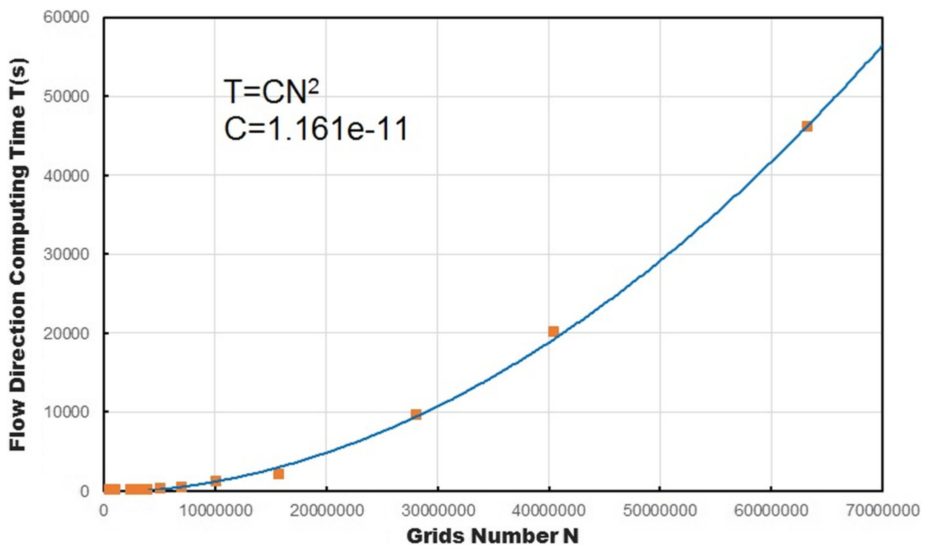

Fig. 9. Computation time curve versus the number of grids.

Table 3
Threshold and sub-basin number.

Resolution (m)		100		200		300		400	
Grid number		80652902		20163399		8961515		5040784	
Threshold (km^2)	Sub-basin number	Calculation time (second)							
		serial	parallel	serial	parallel	serial	parallel	serial	parallel
100	3215	2909	1634	976	567	413	239	337	192
250	1287	1846	980	224	119	132	75	97	54
500	641	1158	628	201	110	93	54	60	35
1000	327	1528	940	236	133	89	52	48	29
2000	155	2367	1445	304	171	104	62	48	28
4000	77	4033	2053	470	239	147	73	58	30
8000	37	13993	10807	1051	752	279	173	104	63
16000	16	26903	22130	1809	1422	426	310	170	135
32000	9	56534	47861	3725	3174	743	637	277	227

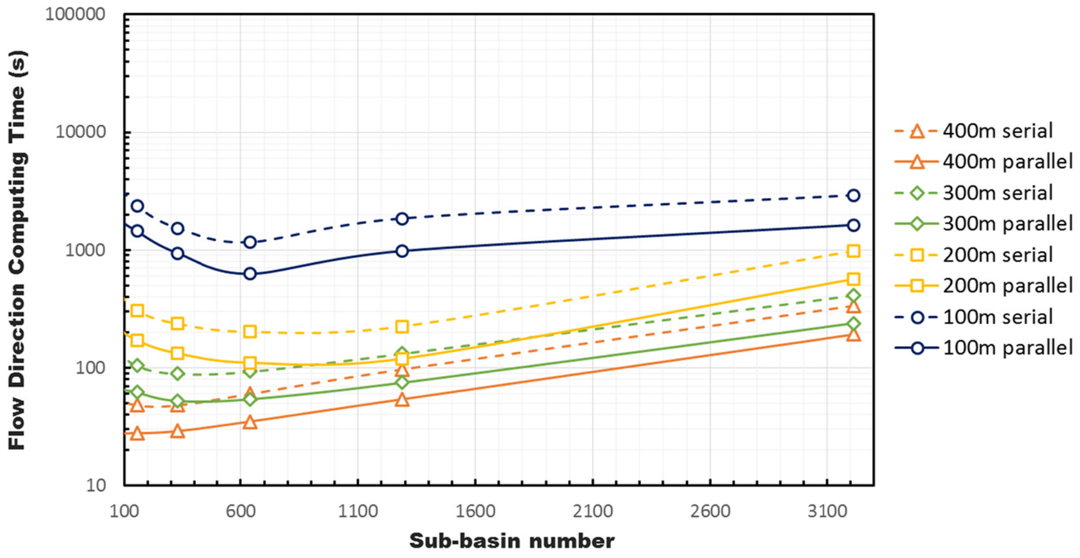

Fig. 10. Flow direction computing time of different thresholds (Y-axis is a logarithmic axis).

Table 4
The results of curve-fitting for both the serial and parallel methods.

Method	a	b	R^2	RMSE
Serial	0.005728	–9.097	0.9453	2604
Parallel	0.004785	–9.579	0.9414	2269

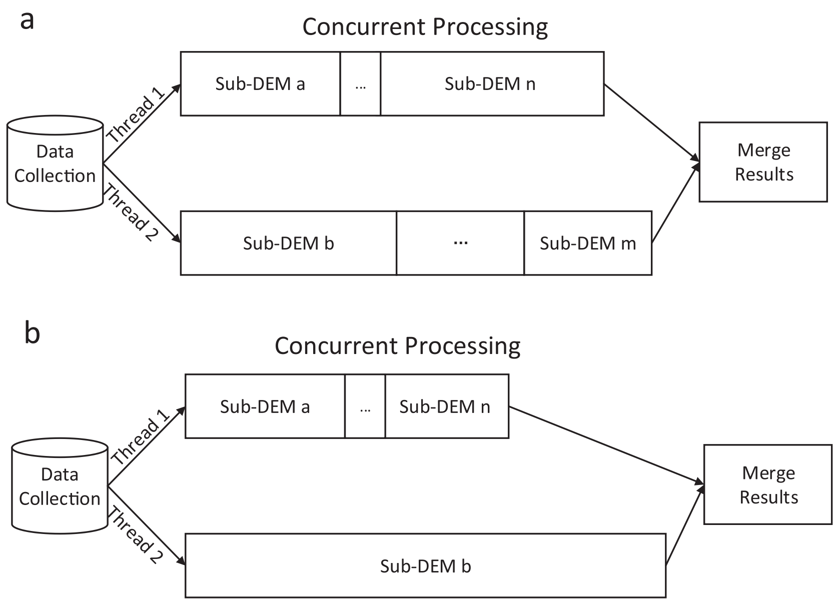

Fig. 11. Different concurrent processing types.

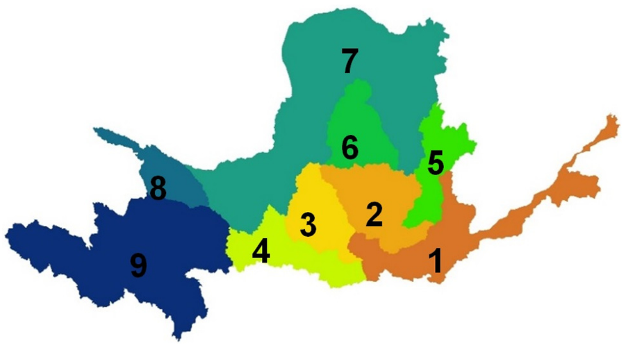

Fig. 12. 32,000 km^2 threshold sub-basin of the Yellow River basin.

5. Conclusions

In this paper, we used parallel programming in a .NET Framework with a C# Compiler to calculate flow direction within a Windows environment. This study resolved the computation problem caused by a large number of grids in a high-resolution DEM to allow personal computers to extract higher-accuracy drainage networks. The method in this study could determine the flow direction from a 100 m resolution DEM, which cannot be calculated directly by a personal computer. On the one hand, the method did not require an excess of computer resources to obtain the high-resolution flow direction, and met the needs of ordinary users. On the other hand, the parallel method also assured a high degree of precision, and the extracted drainage network is highly consistent with the entire extraction.

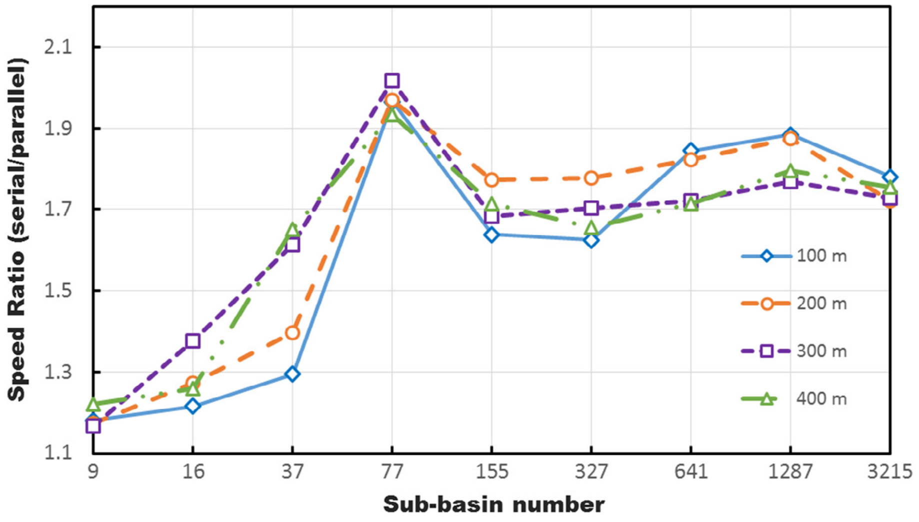

Fig. 13. Speed ratio between serial method and parallel method.

The computation time is not strictly increased with an increase of the threshold value. When there are too many sub-DEMs, the time may reflect a small increase. Comparing the serial and parallel methods, the speed ratio has an average of 1.61 and ranges from 1.1 to 2.1, indicating that the computational efficiency is obviously improved. However, the time difference between the serial and parallel method gets relatively smaller when the number of sub-basin gets smaller. We retrieved the formula for the computing time, grid number and sub-basin number by using curve-fitting, following which the R^2 values exceed 0.94, proving that the formula is reasonable.

Acknowledgments

This study was supported by the Natural Science Foundation of China (No. 41475093), the Intergovernmental Key International S&T Innovation Cooperation Program of China (No. 2016YFE0102400), National Key Research and Development Program of China (No. 2017YFC0404405) and the State Key Laboratory of Severe Weather Open Research Program (No. 2015LASW-A05).

References

Apostolopoulos, T.K., Georgakakos, K.P., 1997. Parallel computation for streamflow prediction with distributed hydrologic models. J. Hydrol. 197 (1), 1–24.
Bai, R., Li, T., Huang, Y., Li, J., Wang, G., 2015. An efficient and comprehensive method for drainage network extraction from DEM with billions of pixels using a size-balanced binary search tree. Geomorphology 238, 56–67.
Clematis, A., Coda, A., Spagnuolo, M., Mineter, M., Sloan, T., 1997. Developing non-local iterative parallel algorithms for GIS on Cray T3D using MPI. Recent Adv. Parallel Virtual Mach. Message Passing Interface, 435–442.
Cui, Z., Vieux, B.E., Neeman, H., Moreda, F., 2005. Parallelisation of a distributed hydrologic model. Int. J. Comput. Appl. Technol. 22 (1), 42–52.
Do, H.T., Limet, S., Melin, E., 2011. Parallel computing flow accumulation in large digital elevation models. Procedia Comput. Sci. 4, 2277–2286.
Garbrecht, J., Martz, L.W., 1997. The assignment of drainage direction over flat surfaces in raster digital elevation models. J. Hydrol. 193 (1), 204–213.
Gong, J., Xie, J., 2009. Extraction of drainage networks from large terrain datasets using high throughput computing. Comput. Geosci. 35 (2), 337–346.
Grimaldi, S., Nardi, F., Di Benedetto, F., Istanbulluoglu, E., Bras, R.L., 2007. A physically-based method for removing pits in digital elevation models. Adv. Water Resour. 30 (10), 2151–2158.
Grübsch, M., David, O., 2001. How to divide a catchment to conquer its parallel processing. An efficient algorithm for the partitioning of water catchments. Math. Comput. Modell. 33 (6–7), 723–731.
Jenson, S.K., Domingue, J.O., 1988. Extracting topographic structure from digital elevation data for geographic information system analysis. Photogramm. Eng. Remote Sens. 54 (11), 1593–1600.
Li, T., Wang, G., Chen, J., Wang, H., 2011. Dynamic parallelization of hydrological model simulations. Environ. Modell. Software 26 (12), 1736–1746.
Li, T.J., Liu, J.H., He, Y., Wang, G.Q., 2006. Application of cluster computing in the digital watershed model. Adv. Water Sci. 17 (6), 841.
Mao, Y., Ye, A., Xu, J., Ma, F., Deng, X., Miao, C., Di, Z., 2014. An advanced distributed automated extraction of drainage network model on high-resolution DEM. Hydrol. Earth Syst. Sci. Discuss. 11 (7), 7441–7467.
Martz, L.W., Garbrecht, J., 1998. The treatment of flat areas and depressions in automated drainage analysis of raster digital elevation models. Hydrol. Processes 12 (6), 843–855.
O'Callaghan, J.F., Mark, D.M., 1984. The extraction of drainage networks from digital elevation data. Comput. Vision Graphics Image Process. 28 (3), 323–344.
Ortega, L., Rueda, A., 2010. Parallel drainage network computation on CUDA. Comput. Geosci. 36 (2), 171–178.
Peucker, T.K., Douglas, D.H., 1975. Detection of surface-specific points by local parallel processing of discrete terrain elevation data. Comput. Graphics Image Process. 4 (4), 375–387.
Poggio, L., Soille, P., 2011. A probabilistic approach to river network detection in digital elevation models. Catena 87 (3), 341–350.
Poggio, L., Soille, P., 2012. Influence of pit removal methods on river network position. Hydrol. Processes 26 (13), 1984–1990.
Qin, C., Zhan, L., 2012. Parallelizing flow-accumulation calculations on graphics processing units—From iterative DEM preprocessing algorithm to recursive multiple-flow-direction algorithm. Comput. Geosci. 43, 7–16.
Richardson, A., Hill, C.N., Perron, J.T., 2014. IDA: An implicit, parallelizable method for calculating drainage area. Water Resour. Res. 50 (5), 4110–4130.
Rueda, A.J., Noguera, J.M., Luque, A., 2016. A comparison of native GPU computing versus OpenACC for implementing flow-routing algorithms in hydrological applications. Comput. Geosci. 87, 91–100.
Sten, J., Lilja, H., Hyväluoma, J., Westerholm, J., Aspnäs, M., 2016. Parallel flow accumulation algorithms for graphical processing units with application to RUSLE model. Comput. Geosci. 89, 88–95.
Tesfa, T.K., Tarboton, D.G., Watson, D.W., Schreuders, K.A., Baker, M.E., Wallace, R. M., 2011. Extraction of hydrological proximity measures from DEMs using parallel processing. Environ. Modell. Software 26 (12), 1696–1709.
Tristram, D., Hughes, D., Bradshaw, K., 2014. Accelerating a hydrological uncertainty ensemble model using graphics processing units (GPUs). Comput. Geosci. 62, 178–186.
Wang, Z., Zheng, H., Liu, C., Wu, X., Zhao, W., 2004. A distributed hydrological model with its application to the Jinghe watershed in the Yellow River Basin. Sci. China Ser. E: Technol. Sci. 47, 60–71.
Ye, A., Xia, J., Wang, G., Wang, X., 2005. Drainage network extraction and subcatchment delineation based on digital elevation model. J. Hydraul. Eng. 36 (5), 531–537.