Computers and Geosciences 109 (2017) 228–237

Contents lists available at ScienceDirect

Computers and Geosciences

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

Research paper

OpenMP parallelization of a gridded SWAT (SWATG)

Ying Zhang[a,b], Jinliang Hou[a], Yongpan Cao[a,b], Juan Gu[c], Chunlin Huang[a,d,*]

[a] *Key Laboratory of Remote Sensing of Gansu Province, Heihe Remote Sensing Experimental Research Station, Northwest Institute of Eco-Environment and Resources, Chinese Academy of Sciences, Lanzhou, 730000, China*
[b] *University of Chinese Academy of Sciences, Beijing, 100101, China*
[c] *Key Laboratory of Western China's Environmental Systems (Ministry of Education), Lanzhou University, Lanzhou, 730000, China*
[d] *Jiangsu Center for Collaborative Innovation in Geographical Information Resource Development and Application, Nanjing, China*

A R T I C L E I N F O

Keywords:
Grid
Parallel
SWAT
Hydrological model
OpenMP

A B S T R A C T

Large-scale, long-term and high spatial resolution simulation is a common issue in environmental modeling. A Gridded Hydrologic Response Unit (HRU)-based Soil and Water Assessment Tool (SWATG) that integrates grid modeling scheme with different spatial representations also presents such problems. The time-consuming problem affects applications of very high resolution large-scale watershed modeling. The OpenMP (Open Multi-Processing) parallel application interface is integrated with SWATG (called SWATGP) to accelerate grid modeling based on the HRU level. Such parallel implementation takes better advantage of the computational power of a shared memory computer system. We conducted two experiments at multiple temporal and spatial scales of hydrological modeling using SWATG and SWATGP on a high-end server. At 500-m resolution, SWATGP was found to be up to nine times faster than SWATG in modeling over a roughly 2000 km^2 watershed with 1 CPU and a 15 thread configuration. The study results demonstrate that parallel models save considerable time relative to traditional sequential simulation runs. Parallel computations of environmental models are beneficial for model applications, especially at large spatial and temporal scales and at high resolutions. The proposed SWATGP model is thus a promising tool for large-scale and high-resolution water resources research and management in addition to offering data fusion and model coupling ability.

1. Introduction

Watershed models have been extensively used by researchers and decision-makers to understand how hydrological and ecological processes, human activities and climate-change affect water resources (Li et al., 2013). SWAT (Soil and Water Assessment Tool) is one of the most widely used watershed models for physically based studies of the Earth and of water resources and environmental management around the world (Arnold et al., 1998). It has been applied for cross-disciplinary research and applications in recent years (Ki et al., 2015; Neitsch et al., 2011; Pai et al., 2012; Zhang et al., 2015). Scholars from around the world have been devoted to developing and applying SWAT.

Numerical models such as SWAT are today not only applied for single scenario tests but also for complex assessments like sensitivity analyses, uncertainty analyses, and data assimilation (Rajib et al., 2016; Xie and Zhang, 2010). Most of these applications require considerable amounts of computational time to use. To save the model users' time spent on model runs or calibrations, a parallel feature (a high-performance computing method) has been successfully integrated into most numerical models (Ki et al., 2015; Li et al., 2011; Vrugt et al., 2006; Yalew et al., 2013; Zhang et al., 2013). From the rapid development of computer technologies, computers now contain many Central Processing Units (CPUs), and each CPU includes several cores (threads) that can execute computational tasks simultaneously. A key issue to consider when programming a parallel model on a distributed or shared memory system involves balancing data across memory and computation units over CPUs (Chapman et al., 2007; Palis et al., 1996). To address this issue, two parallel computing approaches are typically applied when using the SWAT model: (1) Distributed memory parallel (DMP). DMP is a computer system in which each CPU is given its own individual memory. Rouholahnejad et al. (2012) proposed the SWAT-CUP tool for model uncertainty analysis using a Message Passing Interface (MPI) parallel algorithm for DMP. Zhang et al. (2012) developed a software program for optimizing SWAT that can accelerate SWAT optimization by roughly eight-fold on a 16-core

* Corresponding author. Key Laboratory of Remote Sensing of Gansu Province, Heihe Remote Sensing Experimental Research Station, Northwest Institute of Eco-Environment and Resources, Chinese Academy of Sciences, Lanzhou, 730000, China.
E-mail address: huangcl@lzb.ac.cn (C. Huang).

http://dx.doi.org/10.1016/j.cageo.2017.08.002
Received 22 January 2017; Received in revised form 9 July 2017; Accepted 1 August 2017
Available online 2 August 2017

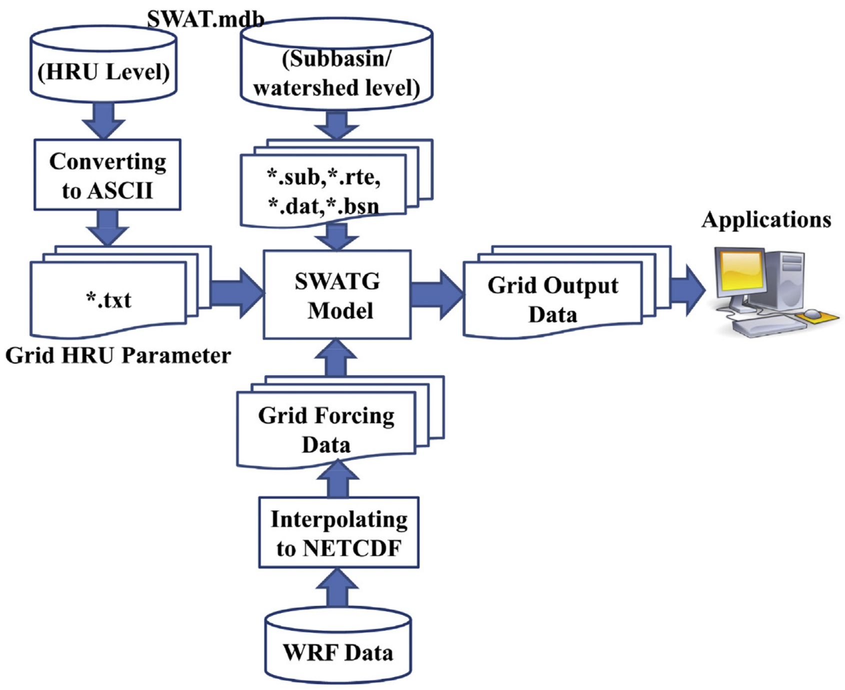

Fig. 1. Schematic flowchart of the SWATG model application.

SMP system. Though the efficiency of the calibration procedure has in turn been improved, SWAT modeling time was not decreased through the above studies. Wu et al. (2013) employed MPI to parallelize SWAT based on subbasin level to speed up individual model runs and manual and automatic calibration procedures and to optimize best management practices on a DMP system. (2) Shared memory parallel (SMP). SMP is a system through which all CPUs can access the same memory and benefit from low-latency memory-access while sharing data among CPUs during computation. OpenMP (Open Multi-Processing) is one of the most suitable APIs for software parallelization on an SMP system. The use of OpenMP for enhancing the computational efficiency of a single SWAT run (especially for parameter calibration procedures) has been previously discussed (Ki et al., 2015). OpenMP is highly scalable for model applications. Each hydrological model must be employed through a distinct parallel computing strategy and based on its own modeling structure.

The objective of this paper is to describe the parallel method (OpenMP) as it is implemented in the gridded SWAT (SWATG). We compare the computation time of the parallel SWATG by applying it to study a basin with at different spatial resolutions over different periods using different computer configurations. Finally, we summarize the performance of the parallelized SWATG and computation time saved through parallel processing. Source codes of SWATG and SWATGP are listed on the first author's website: http://hexiayouxi.tk.

2. Overview of gridded SWAT

Three spatial modeling levels are used in SWAT: the watershed, subbasin and Hydrologic Response Unit (HRU) levels. The model presents weaknesses in terms of subbasin level hydrologic modeling. Spatial variability in weather conditions within a subbasin cannot be adequately represented by a weather station (forcing data have a uniform distribution at the subbasin level). This spatial distribution of hydrologic components is not natural and significantly affects the accuracy of watershed hydrologic process modeling (Domeneghetti et al., 2014). Another weakness related to SWAT concerns its delineation of HRUs. HRUs with irregular shapes and no geospatial locations complicate model coupling, data fusion, manager implementation and BMP analysis. Modeled results (e.g. soil moisture and snow cover) are difficult to map for 2D visualization. Traditional SWAT model input and output data integrate a combination of raster, vector and tabular data with complex structures, restricting their incorporation by other tools. Additionally, more and more data are now accessible and most of them are of gridded forms (e.g., remote sensing data) and can be used through SWAT to improve modeling accuracy levels and our understanding of watersheds (Chen et al., 2011).

Grid modeling schemes are advantageous in that the integration of gridded data reveals spatial characteristics at the represented resolution (Liang et al., 2004). Such models based on regular grids such as VIC (Liang et al., 1996) can provide spatial distribution characteristics of various critical water cycle components for regional and global earth science studies. The impacts of spatial changes in land use and BMPs on specific regions can be more realistically assessed through a gridded model (Arnold et al., 2010). Usually, to achieve a comprehensive physical representation, an individual model with or without poor physical process representation must be extended (e.g., to couple with other models). Several valuable models (e.g., WRF, MODFLOW) which have hardly been coupled with SWAT but can be efficiently coupled with grid-based models. A gridded SWAT model based on subbasins has been proposed as a means to address the problems we noted above (Rathjens and Oppelt, 2012; Rathjens et al., 2015). However, the number of input files for such a SWAT project increases dramatically as the number of delineated subbasins increases. The approach cannot be applied for large-scale high-resolution watershed hydrologic modeling. Modeling, data processing and storage will cost model users a great deal of time and effort, in turn will inhibiting the development and application of the model. A SWAT model with not only grid features but also offering computationally efficient implementation can facilitate watershed research and application.

3. Materials and methods

3.1. *The gridded soil and Water Assessment Tool (SWATG)*

The Soil and Water Assessment Tool (SWAT) provides detailed representations of physical processes related to water and nutrient movement such as percolation, evapotranspiration, runoff, groundwater

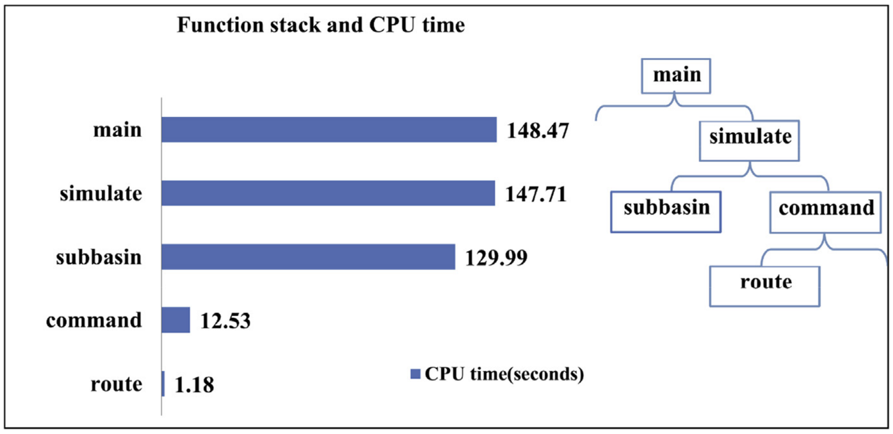

Fig. 2. Execution time of the functions' tree (only five most time-consuming subroutines are shown) in the SWATG model for a simulation period of one year.

system, nutrient cycling, and pollution transport (Arnold et al., 1998). A watershed is divided into several sub-watersheds (subbasins) based on Digital Elevation Model (DEM) data, and each subbasin is composed of Hydrologic Response Units (HRUs) with uniform soil characteristics, land use types, and topological characteristics (slopes). Hydrological processes such as evaporation, runoff, infiltration, lateral flow, irrigation, snowmelt and ground water systems are modeled in each HRU. In most cases, an HRU with a certain soil type, land use type and slope often occupies several discrete grids in which the water balance is simulated independently without interactions within one subbasin. Subbasin-level variables are summed up by modeled HRU-level variables. After subbasin level simulation is completed, the model routes water and other elements upstream to downstream, i.e., from the entire watershed to its outlet through the channel network. Input observational weather data include daily/hourly precipitation, air temperature, wind speed, solar radiation and relative humidity data. The water balance for each HRU is calculated as follows:

$$SW_{\mathrm{m}} = SW_o + \sum_{j=1}^{\mathrm{m}} \left(P_{\mathrm{j}} - Q_j - E_{a,j} - S_j - G_j\right) \tag{1}$$

where SW_{m} is the soil water content at time m; SW_o is the initial soil water content; and P_{j}, Q_j, $E_{a,j}$, S_j and G_j are precipitation, surface runoff, actual evapotranspiration, infiltration and groundwater flow at time j, respectively. All units are recorded in mm H2O.

We modified SWAT to consider each grid cell as an HRU (also referred to as a gridded HRU; the modified model can be referred to as a gridded HRU-based SWAT (SWATG)). Therefore, HRU-related parameter variables (e.g., slope length) must be redefined. HRU has an additional characteristic: spatial locations within the watershed grid map. Subbasin loops are nested in HRU loop simulation in the SWATG due to independent modeling between subbasins. The subroutine **virtual** is modified to sum HRU variables to the subbasin level at the end of each HRU loop. Fig. 1 shows components of the SWATG model architecture.

Input data for SWATG include grid forcing data, soil parameters (e.g., available water content and the saturated hydraulic conductivity of soil layers), land surface parameters (e.g., plant water update compensation factor, Manning's "n" value for overland flow, and lateral subsurface flow slope length) and HRU identification numbers. Grid forcing data can be obtained through remote sensing, weather modeling or weather station data interpolation. Grid parameters can be generated via the ArcSWAT software extension based on land cover, soil map and DEM data for simplification (used in this study). Each HRU has unique meteorological inputs (rather than sharing nearby weather station data within one subbasin in SWAT). Control file ("file.cio") including the spatial and temporal configurations for the simulation has also been adapted to the SWATG.

3.2. *Hierarchy of SWATG subroutines and OpenMP implementation*

OpenMP is an Application Program Interface (API) that can be used to explicitly direct multi-threaded, shared memory parallelism for programs that can be used in major programming languages such as C/C++, FORTRAN and Java using thread libraries (Quinn, 2005). It is a set of compiler directives that can enhance loop-level parallelism for parallel programming via SMP systems. In this study, OpenMP parallel method is integrated into SWATG to address the model's computational efficiency problems. The shared variables' determination in the SWATG model is a critical component for parallelization. When using OpenMP to implement shared memory parallelization, we must overcome two major challenges: 1) identifying and parallelizing suitable parts of the code; and 2) avoiding data races due to parallel updates of shared variables.

First, we must identify the most needed improved function (subroutine) through SWATG. Some works have used the GNU gprof tool, a performance analysis tool for Linux applications, to identify the slowest subroutines in applications (Artes et al., 2015; Graham et al., 1983, 2004; Ki et al., 2015). The manual for gprof can be found at https://sourceware.org/binutils/docs/gprof/. We created a SWATG with option "-pg" for memory profiling and found that the most time-consuming subroutine within the **simulation** subroutine is **subbasin** (Fig. 2). The modified **command** subroutine controls the routing phase, including routing water and sediment travelling to the main channel after subbasin simulation. The **subbasin** subroutine, which is excluded from the original **command** subroutine, controls the HRU loop, and so the time dedicated to the **subbasin** subroutine is the total amount of HRU loop execution time needed. The HRU loop controls the water balance simulation executed in SWATG, including variable initialization (subroutine **varinit**), rainfall-runoff processes (subroutines **surface** and **percmain**), evapotranspiration (subroutines **etact** and **etpot**), plant growth (subroutine **plantmod**), sediment and nutrient generation and movement (subroutine **nutrient**), and ground water process (subroutine **gwmod**). Subroutines **sumv** and **virtual** are used to sum HRU level variables. Based on this evaluation of SWATG, the HRU loop is the most critical component in terms of computation costs. The SWATG model is parallelized at the HRU level as HRU simulation is conducted independently and the HRU loop (the most time-consuming procedure) must be made more efficient.

Then, variable initialization and HRU simulation in related subroutines are modified to include the necessary OpenMP components. The OpenMP directives used in this study mainly contain OMP PARALLEL THREAD PRIVATE, FIRST PRIVATE, PARALLEL ATOMIC and PARALLEL DO structures. THREAD PRIVATE is used to initialize and define global variables that cannot be shared by the same memory address. FIRST PRIVATE is used to initialize and define local variables that cannot be at

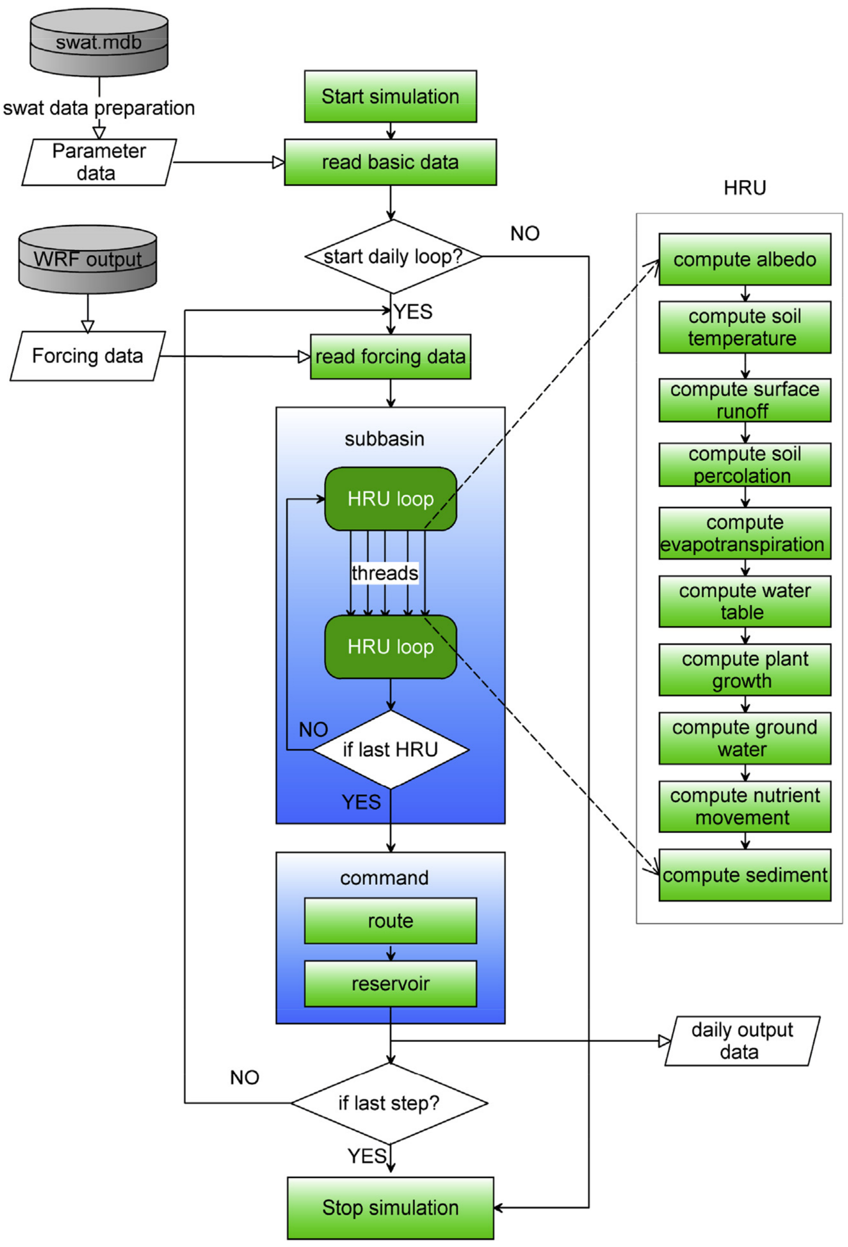

Fig. 3. Representation of the structure of SWATGP. Black arrows indicate individual computational threads.

the same memory address within specific computational regions. The PARALLEL ATOMIC directive controls global variable updates or recursion. PARALLEL DO is the most important operation because it distributes the HRU loop over computational threads to improve loop efficiency levels. Parameter declaration and initialization subroutine ("modparm.f") is modified to contain the OMP THREAD PRIVATE directive to redefine global variables as private. For example, each HRU contains a memory address of *inums* that is unique for each HRU in SWATG. The PARALLEL DO Directive is added to the HRU loop structure. The PARALLEL ATOMIC directive is added to subroutines where global variables must be updated atomically rather than multiple threads to attempt to write to it. At this point, each HRU simulation can be simultaneously performed on individual threads. Once an HRU loop simulation is complete, hydrological variables of the HRUs are summed to the subbasin level or to the outlet of a subbasin. In this context, improving the efficiency of SWATG through parallel computing is critical for practical fine spatial resolution (HRU) watershed modeling.

Finally, the SWATG routing function is performed on the main thread to route water from upstream to downstream through reaches. The sequential variables (e.g., subbasin runoff) input for routing phase simulation are dependent on one another, and so they are difficult to parallelize. The basic structure of the SWATGP is illustrated in Fig. 3. The parallel implementation of SWATG is straightforward when using the GNU library. When compiling SWATGP with the GNU compiler, the "*-fopenmp*" flag is needed. A brief example of the modification codes is provided in Appendix. Further information on the OpenMP parallel method can be found at http://www.openmp.org.

3.3. Evaluation method

In this work, we complied the SWATG program using the gfortran compiler, ran the standard simulation in the serial mode, and stored output files in a separate directory. All serial output files were designated as reference results to be compared with those of SWATGP. We used an internal function **system_clock** to measure the time spent on SWATG and

Table 1
Computer configuration used for model simulations.

Item	Description
CPU	Intel(R) Xeon(R) CPU E7- 4850 @ 2.00 GHz
Logical cores	80
Physical cores	40
CPUs	4
RAM	64 G
System type	Linux 64-bit CentOS 6.5
Compiler	GNU4.4.7 and Python3.4.1

SWATGP execution. We added several FORTRAN calls as **system_clock** in the **subbasin** subroutine to measure the execution time of HRU loop. The following performance measure is commonly used to evaluate the performance of parallel computations: speedup. Speedup for *n* parallel sessions is defined as the ratio of the computed time dedicated to a task when only one core is used to the computed time when *n* cores are used (Ki et al., 2015).

The underlying architecture of OpenMP can be shared memory Uniform Memory Access (UMA) or Non-Uniform Memory Access (NUMA). Under UMA, all processors share the same amount of physical memory. Under NUMA, a processor can access its own local memory faster than non-local memory. Thus, the parallelized high-performance SWATG with can benefit from specific memory and thread placement and optimized memory locality (Kiefer et al., 2013). Different CPUs and logical cores in our computer were employed to evaluate the NUMA effect on parallel efficiency. We carried out experiments with SWATGP using combinations of 1 CPU, two CPUs, three CPUs and four CPUs and threads of 1–80. The Linux numactl tool was used to combine CPUs and threads and to allocate memory among all CPUs (Kiefer et al., 2013). The manual for numactl can be found at https://linux.die.net/man/8/numactl.

4. Case study and evaluation

4.1. Study area and data preparation

The Babaohe River, a tributary of the Heihe River originating from the Qilian Mountains, flows through the county of Qilian in the arid region of northwestern China. The Babaohe basin covers an area of 2455 km^2 and is positioned at 2678 m to 4883 m above sea level. Snow and ice melt serve as the most important sources of water in this region during the spring. The mean annual precipitation of the area is 400 mm, whereas potential evapotranspiration is exceed 600 mm. The dominant vegetation types include alpine meadows, sub-alpine shrubbery, and alpine steppe.

The Babaohe Basin SWAT project was first initiated with the ArcSWAT extension of ArcGIS 10.0 software. Input datasets for the SWAT project were acquired from the WestDC (http://westdc.westgis.ac.cn/) online open-access resource, including the following required input data for SWAT: digital elevation model (DEM) data, soil texture data (Harmonized World Soil Database, HWSD) and a land use map (WESTDC). Meteorological station data were extracted from gridded forcing data and used as climate inputs and weather generator for SWAT. Gridded forcing data (rainfall, air temperature, solar radiation, humidity and wind) were generated from the WRF model (Pan et al., 2015). In the sub-watershed discretization scheme, subbasins were delineated based on DEM data. The spatial resolution of the grid (HRU) was set to 500 m × 500 m (250 m × 250 m was also used to evaluate the effects of spatial scales). After being gridded, the Babaohe Basin was found to contain 9707 HRUs, which is substantially greater than the original 654 HRUs delineated from ArcSWAT based on unique soil types, land use types and terrain characteristic (slopes). The gridded parameters (e.g., available water content, soil hydraulic conductivity, soil bulk density, soil albedo, soil organic carbon and crack volume, slope steepness, land cover classification, HRU and subbasin identification numbers) were prepared by converting the original SWAT project input data into gridded data using ArcGIS lookup and conversion tools. Parameters of configure and control files ("file.cio" and "basins.bsn") for the SWATG versions were also set for specific applications.

4.2. Computing environment and software description

A high-performance computer (high-end server) with multiple CPUs or logical cores was used for model simulations. As stated in Section 3.1, input data were prepared using ArcSWAT and Python software on a personal computer. The high-end server's configuration details are described in Table 1. The logical cores can be thought of as threads, which are the finest computation units in a computer. The high-end server used in this study included four CPUs and each CPU included 10 physical cores and 20 logical threads. The OpenMP parallel interface was enabled. A GNU compiler was used as a FORTRAN language compiler, and Python was used as a data processing tool to transform SWAT input data into SWATG/SWATGP input data. It was necessary to install the NetCDF FORTRAN and Python libraries for the present study.

SWATG software versions (SWATGs) used in this study included the following:

(1) SWATG: a modified SWAT model based on SWAT2012 that treats a grid cell as an HRU. The difference between the original SWAT model and this modified model lies in its ability to accommodate gridded forcing and parameter data.
(2) SWATGP: a modified SWATG model that can run on a multi-core computing platform through the OpenMP parallel application interface.

4.3. Parallel assessment of SWATGP

4.3.1. Experiment setup

To evaluate the simulation performance of SWATGP, we executed the same Babaohe Basin modeling project with two separate models: SWATG and SWATGP.

Parallel efficiency: We evaluated parallel efficiency levels of SWATGP through the Babaohe Watershed modeling project relative to those of SWATG. First, the time step was set as daily, and the temporal range extended from January 1st to December 31st of 2008 for model evaluation and subsequent research. Then, the Babaohe Basin project was executed using versions of SWATG and the gridded data derived from the WRF model (Pan et al., 2015). Each model run time was exported for subsequent comparison. The run time of each subroutine within **subbasin** was also exported for both models.

Grid effect on parallel: To assess the grid method's effect on modeling efficiency levels, we employed SWATGs for different spatial, temporal and parallel configurations. Two Babaohe Basin projects were executed using HRU resolutions of 250 m × 250 m and 500 m × 500 m to demonstrate and evaluate the influence of the number of HRUs on the model's performance. There were a total of 9707 and 97,240 HRUs in the hydrological simulations of the 500 m × 500 m and 250 m × 250 m resolutions, respectively. The time step was set to one day, and the execution time of the HRU loop for each day was calculated as a daily average over each time step. The total model execution times for the one-, two-, five- and ten-year simulations were computed. NUMA effects were also evaluated for different SWATGP resolutions.

4.3.2. Results and discussion

Output files including grid outputs (e.g., soil moisture, runoff and evapotranspiration) simulated from SWATG and SWATGP were compared for modification validation. We found no differences between the SWATG and SWATGP simulation results for the same forcing data and employed the diff tool to ensure this conclusion. Input/output file reading/writing processes, which largely depend on computer hard disks and memory sizes, are not our focus. The main modifications of SWATG pertain to HRU-level water balance processes modeling specifications.

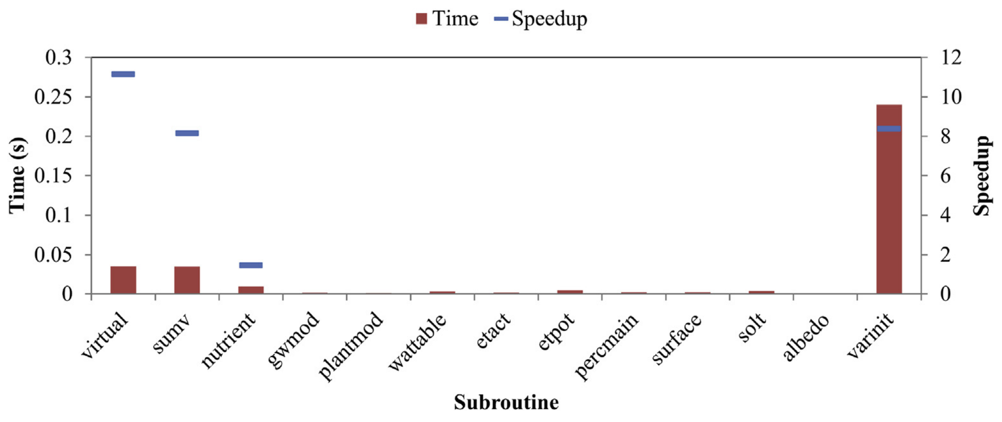

Fig. 4. Serial run time (Time) and speedup results for subbasin subroutines.

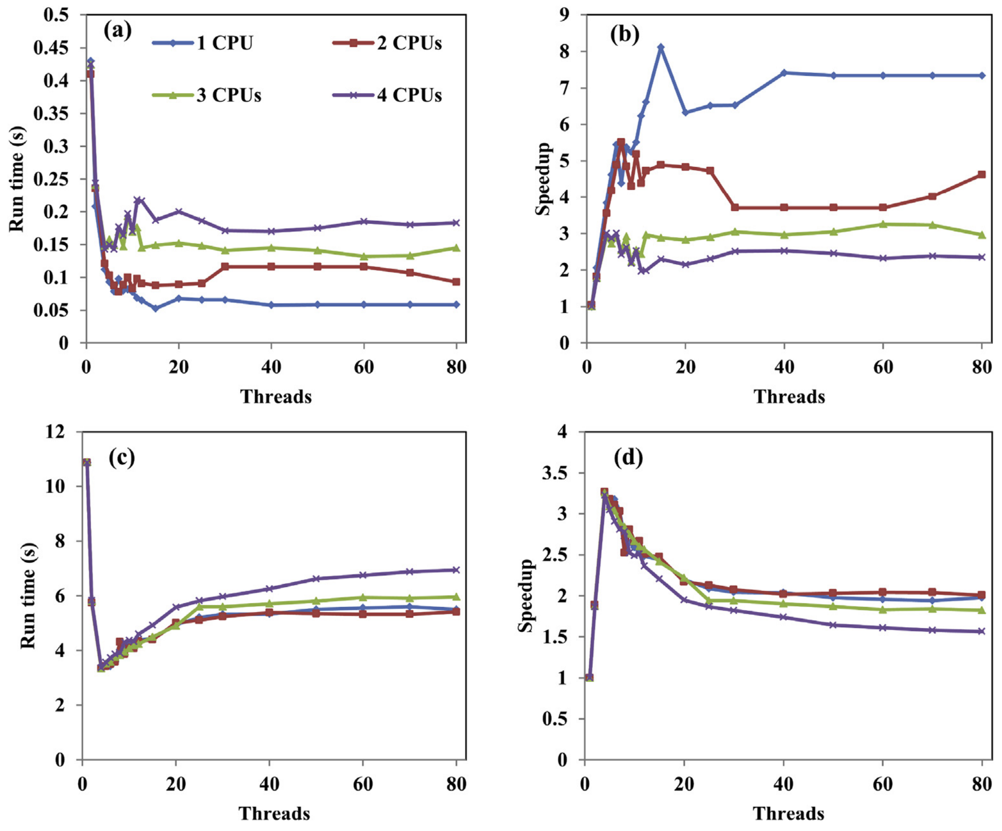

Fig. 5. Run time and speedup for daily SWATGP subbasin subroutine for different resolutions (500 m (a, b), 250 m (c, d)) based on different combinations of CPUs and threads.

The other modeling phase (e.g., routing phase) is not modified and is not affected by parallel implementation. We thus illustrate the parallel computing performance of SWATGP only in relation to HRU simulation processes (**subbasin** subroutine).

Different subroutines of **subbasin** may have different effects on the speedup results. Fig. 4 shows the execution time for subroutines of the **subbasin** and their speedup outcomes resulting from parallel implementation. As only one CPU (10 physical cores) is employed in this case, a desirable speedup through parallel implementation can approach 13 when Hyper-Threading (two threads executed on each core) is enabled. Because only up to 30% gains can be achieved using a certain programs with Hyper-Threading enabled (Hennessy and Patterson, 2011; Hill and Marty, 2008; Patterson, 2003). We also tested Hyper-Threading effects of SWATGP and found that the speedup results were not significantly improved when Hyper-Threading was enabled. The most time-consuming subroutine within **subbasin** was found to be **varinit**, which initializes HRU variables at the start of the HRU loop. The speedup for **varinit** is also impressive in its parallel use within **subbasin**. Subroutines **virtual** and **sumv** also take a considerable amount of time within **subbasin** and can be improved considerably through parallel implementation. The **nutrient** subroutine includes many synchronization processes (e.g., variable updating with atomic instruction) that can dramatically affect parallel efficiency levels. Other water balance modeling subroutines (e.g., **gwmod** and **etact**) already cost little time, and their speedups via parallel implementation can be disregarded and are thus not shown here. There are 32 and 136 atomic instructions in **virtual** and **sumv,** respectively. As the number of synchronized variables increases, the speedup for the subroutine decreases. HRU variable initialization and synchronization are key processes that affect parallel efficiency levels. However, initialization and synchronization processes are difficult to

Table 2
Simulation settings and performance of the SWATG and SWATGP models.

Project resolution	HRUs (grid cells)	Run time of the daily **subbasin** subroutine (seconds)	
		SWATG	SWATGP
500 m × 500 m	9707	0.43	0.05 (1 CPU, 15 threads)
250 m × 250 m	97,240	11.2	3.31 (1 CPU, 4 threads)

speed up. Using thread-local storage variables rather than synchronized variables can further speed up parallel efficiency levels. While there are several synchronized variables in **subbasin** whose dimensions are determined by the number of subbasins and/or HRUs involved, converting them to thread-local is not simple and should be studied at more length.

As noted above, a program run on the NUMA system must be carefully managed through the use of specific threads and memory placement policies. Based on unique combinations of CPUs and threads, the run time and speedup results for the daily SWATGP at different resolutions can be seen in Fig. 5. The run time steadily declines until more than 15 threads (for a 500-m resolution) or 4 threads (for a 250-m resolution) are used (See Fig. 5). We find that the best parallelization for a 500-m SWATGP can be achieved when using 1 CPU and 15 threads and when using 1 CPU and 4 threads for a 250-m SWATGP. These results can be attributed to thread migration and memory allocation problems. The amount of memory allocated for a 500-m SWATGP is less than that for a 250-m SWATGP as shown in our other findings (Fig. 5). In turn, less time is required for thread migration for a 500-m SWATGP than for a 250-m SWATGP. Thus, more threads can be used to accelerate a 500-m SWATGP

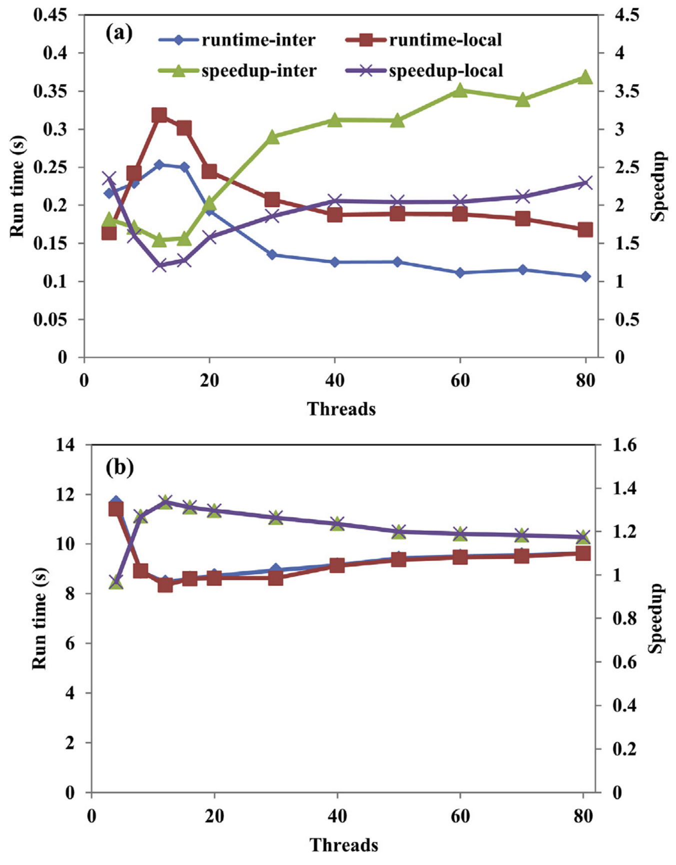

Fig. 6. Run time and speedup results for daily SWATGP subbasin subroutine conducted at 500-m (a) and 250-m (b) resolutions using interleaving and local allocation memory allocation strategies (inter and local) and threads.

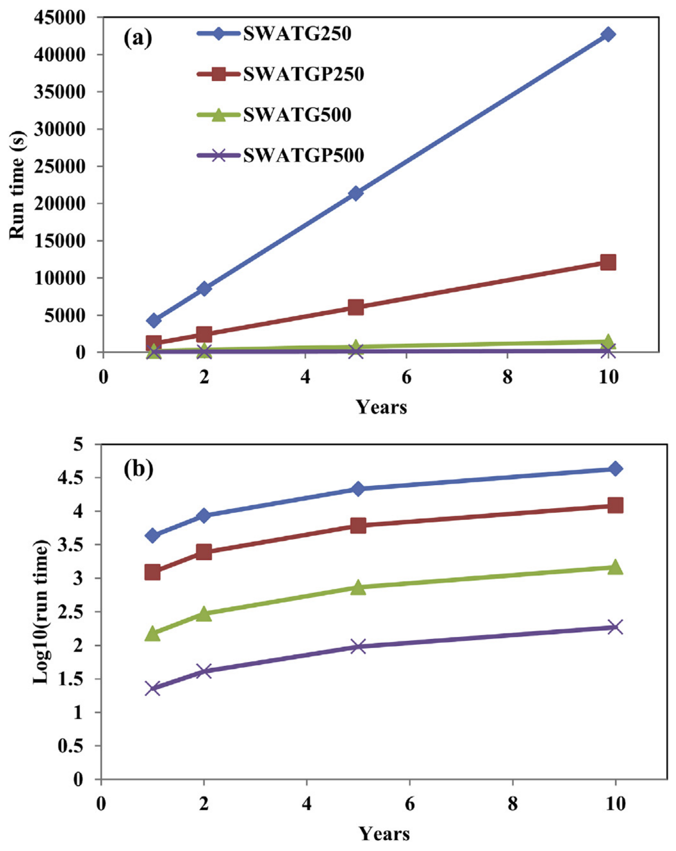

Fig. 7. Run time (a) and Log10 (run time) (b) for SWATG and SWATGP based on different simulation durations and at different grid resolutions (250 m × 250 m and 500 m × 500 m).

than a 250-m SWATGP through faster communication. When more than 4 threads are involved (or 15 threads), the parallel overhead outweighs the gain from parallel computations. In the default case (without numactl), threads of all instances are applied to all sockets, and they frequently migrate between CPUs, spurring fluctuations in run time series. Table 2 shows the results of SWATG and SWATGP based on applications of best NUMA scheduling strategies. With 1 CPU and 15 threads, the speedup of SWATGP subbasin subroutine is roughly 8.6-fold at a 500-m resolution. With 1 CPU and 4 threads, the speedup of SWATGP subbasin subroutine is roughly 3.4-fold at a 250-m resolution (Table 2). When the thread count is higher/lower than a threshold, (e.g., 15 threads for a 500-m SWATGP), the HRU loop run time decreases/increases as the number of computational threads increases.

As the amount of memory allocated and thread migration processes affect model performance, we carried out experiments using interleaving and local memory allocation strategies and different threads for SWATGP at 500-m and 250-m resolutions. Corresponding results are shown in Fig. 6. For an interleaving memory policy, memory is allocated using round robin on CPUs. Memory for threads can be allocated from the current CPU or from other CPUs. As part of a local memory policy, SWATGP only allocates memory from certain fixed CPUs. The time required for remote memory access, thread migration and parallel computation should be considered. For a 500-m resolution (Fig. 6a), the runtime for interleaving and local memory allocation increases with an increasing number of threads until 12 threads are used. When more than 12 threads are used, the run time of SWATGP steadily declines for both strategies. Thread migrations take more time than parallel computations when the number of threads involved is less than 12. When only four threads (one thread for each CPU) are used, relatively more execution time is required to access remote memory than for parallel computation;

the local strategy thus is better than the interleaving strategy. When more than four threads are used, employing interleaved memory allocation for parallel computation is better than employing the local strategy. For a 250-m resolution (Fig. 6b), the runtime for interleaving and local memory allocation decreases with increasing number of threads until 12 threads are used. When more than 12 threads are used, the run time of SWATGP slightly increases for both strategies. The interleaving strategy performs nearly the same as the local strategy. When the amount of memory allocated is relatively large (the HRU number is large), memory allocation is distributed uniformly and either interleaving or local strategies can be used. When less memory and fewer threads are allocated, a local strategy is sufficient. When numerous threads are involved, memory resources can be better allocated via an interleaving strategy. For a specific program used on a specific computer, thread migration and memory allocation strategies should be balanced to achieve high levels of computing performance.

Execution times for simulation periods of SWATG and SWATGP at different grid resolutions (250 m × 250 m and 500 m × 500 m) are shown in Fig. 7. The execution time for the one-year SWATG simulation is slightly greater than that of SWATGP. However, according to a ten-year simulation, the execution time for SWATG is dramatically higher than for SWATGP. For example, the amount of execution time required for one-year SWATG and SWATGP modeling at 250 m (SWATG250 and SWATGP250) is roughly 4200 s and 1200 s respectively. The execution time required for ten-year SWATG250 and SWATGP250 modeling is roughly 42,000 s and 12,000 s respectively. Parallelization saved roughly 3000 s and 30,000 s for one-year and ten-year modeling respectively. More time was saved as the simulation period extended. As the simulation period was extended, SWATGP became more efficient than SWATG (Fig. 7).

SWATGP simulation took substantially less time to complete than SWATG simulation, and the amount of time required decreased as the number of computational cores used increased. As the parallel scheme is based on the HRU level involved, when the number of HRUs is increased in a watershed as a result of finer discretization, the amount of time required for HRU modeling also increases. As the simulation time is extended, the advantages of SWATGP become more pronounced than those of SWATG (Fig. 7). The above results show that the introduction of parallelization is valuable for large-scale high-resolution hydrological modeling. The SWATGP begins as a single process (master thread) and is executed sequentially until the parallel region construct (HRU loop) is encountered (see Fig. 3). SWATGP parallel region initialization takes a considerable amount of time in a parallel run time when more than 2000 global variables must be initialized. In this study, we used the SWATG model without HRU routing, which means there was no dependence between the HRUs but that their sum represents subbasin-level characteristics. The HRU variable summation process (a nested-loop) has limited effects on the performance of the parallel procedure.

The performance of a parallelized model can be affected by the number of cores (threads), the cache size, the memory bandwidth, and networking topologies (Wang et al., 2013). The CPU used in this study includes 10 physical cores, and each core has two threads when Hyper-Threading is enabled (Table 1). Programs can only achieve 30% performance improvement from Hyper-Threading on an Intel Westmere-EP processor (Saini et al., 2011). Our results show that the best speedup results are achieved when using 1 CPU and 15 threads with 56% speedup efficiency (speedup per thread) with Hyper-Threading on. SWATGP did not benefit considerably from Hyper-Threading. Given a fixed amount of computing resources, the hybrid parallel mode may be preferred to the pure MPI and OpenMP modes when using a cluster platform with a large number of high-end servers. A symmetric multiprocessing (SMP) cluster that has been widely used in recent years is used at our research institution and will be used as a platform for our future research.

When applying the model (SWATG and SWATGP) reading procedure, 500-m resolution models take 7.48 s to perform and 250-m resolution models take 29.46 s to perform. As the higher resolution increases, more file reading time is required. Output files output time is not considered here due to the optional outputting attributes of output files. Input file reading issues should be considered when very high spatial resolution results are required. Long-term modeling can neglect reading procedure time when the modeling time is dramatically longer.

5. Summary and outlooks

As a modified SWAT model that integrates grid forcing and parameter datasets, SWATG can generate more detailed hydrological process information. The SWATG model can serve as a valuable tool for water resource research and applications based on flexible input/output formats and spatial attributes. The time intensive nature of the model limits its use for long-term and high-resolution simulations. When the spatial resolution is high, the time-consuming nature of hydrological modeling is more evident. In this study, a parallel tool called OpenMP API was incorporated into SWATG to improve the model's computational efficiency. The results show that SWATGP largely reduced the amount of computational time required for the simulations relative to SWATG. There are roughly 2000 shared variables **subbasin** subroutines. As the number of synchronized variables increases, **subbasin** subroutine acceleration declines. Variable initialization and synchronization processes govern the parallel efficiency of SWATGP. Thread-local variables can be used to improve the parallel efficiency of SWATGP, but this capacity is constrained by the number and dimensions of shared variables involved. NUMA effects observed for SWATGPs of different resolutions were also assessed too. At resolutions of 250 m and 500 m, SWATGP was respectively found to be up to three and nine times faster than SWATG in modeling a 2000 km^2 watershed for 1 CPU and 15 threads and for 1 CPU and 4 threads. As the amount of simulation time increases, SWATGP presents more obvious advantages.

Overall, the SWATGP presented here makes a unique contribution to the existing SWATG model. With its capability to save time, SWATGP can help SWAT users study and manage a watershed. As parallel implementation can reduce the execution time of an individual model run, the time required for model calibration can also be shortened. In anticipation of the further development of computer technology (e.g., CPUs), it is worth studying SWATG parallelization strategies regardless of hardware limitations involved (e.g., memory and hard disk limitations) (Rouholahnejad et al., 2012). Different quantities of CPU cores used in SWATGP can result in different levels of acceleration (Liu et al., 2016). This is largely dependent on the levels of communication between CPU threads (CPU structures) (Ki et al., 2015). We believe that the data transferring between CPU sockets constrains the use of more CPUs for speeding up SWATGP. When the memory usage for the model beyond the capable of a single CPU, the data transferring between CPUs is not optimized by OpenMP. We thus plan to develop an MPI/OpenMP hybrid parallel inference method for the SWATG model. With HRU routing integrated, SWATG can also be parallelized through specific parallel implementation methods (e.g., the method proposed by Burger et al. (2014) and Liu et al. (2016)). With the rapid development of computer and remote sensing technologies, opportunities to use SWATGP for long-term, high-resolution hydrological modeling are promising.

Acknowledgements

This work was supported through the National Science Foundation of China under Grants 91425303 and 41671375 and the Science and Technology Service Network Initiative Project of the Chinese Academy of Sciences. We thank the anonymous reviewers for providing valuable suggestions and careful reviews to improve our work.

Appendix

1) Declaration of thread private variable in module.

```
!$omp threadprivate (ep_day, inum1, inum2, inum3,…
!$omp& …ubntess)
```

2) Parallel HRU loop algorithm.

```
!$omp parallel do
!$omp& default (shared)
do j=1,mhru
    call varinit ! hru variable initialization
    call surface
    …
    call virtual ! sum of hru variables for subbasin variables
end do
!$omp end parallel do
```

3) Variable update.

```
! HRU-loop
do j=1, mhru
!$omp atomic
! nested subbasin loop, i.e. precipitation and water yield updating
do i=1, subtot
    sub_subp(sb) = sub_subp(sb) + subp(j) * hru_fr(j)
    sub_wyld(sb) = sub_wyld(sb) + qdr(j) * hru_fr(j)
end do
end do
```

j: hru number

mhru: maximum hru number

sub_subp: subbasin level precipitation

sb: subbasin number

subtot : maximum subbasin number

subp: precipitation for hruhru_fr: fraction of subbasin area contained in hru

sub_wyld: subbasin level water yield which will be routed to the outlet of the basin through routing process.

qdr: hru level water yield

References

Arnold, J.G., Allen, P.M., Volk, M., Williams, J.R., Bosch, D.D., 2010. Assessment of different representations of spatial variability on swat model performance. Trans. Asabe 53, 1433–1443.

Arnold, J.G., Srinivasan, R., Muttiah, R.S., Williams, J.R., 1998. Large area hydrologic modeling and assessment - Part 1: model development. J. Am. Water Resour. Assoc. 34, 73–89.

Artes, T., Cencerrado, A., Cortes, A., Margalef, T., 2015. Enhancing computational efficiency on forest fire forecasting by time-aware genetic algorithms. J. Supercomput. 71, 1869–1881.

Burger, G., Sitzenfrei, R., Kleidorfer, M., Rauch, W., 2014. Parallel flow routing in SWMM 5. Environ. Model. Softw. 53, 27–34.
Chapman, B., Jost, G., Pas, R.V.D., 2007. Using OpenMP: Portable Shared Memory Parallel Programming (Scientific and Engineering Computation).
Chen, F., Crow, W.T., Starks, P.J., Moriasi, D.N., 2011. Improving hydrologic predictions of a catchment model via assimilation of surface soil moisture. Adv. Water Resour. 34, 526–536.
Domeneghetti, A., Tarpanelli, A., Brocca, L., Barbetta, S., Moramarco, T., Castellarin, A., Brath, A., 2014. The use of remote sensing-derived water surface data for hydraulic model calibration. Remote Sens. Environ. 149, 130–141.
Graham, S.L., Kessler, P.B., Mckusick, M.K., 1983. An execution profiler for modular programs. Software-pract. Exp. 13, 671–685.
Graham, S.L., Kessler, P.B., McKusick, M.K., 2004. Gprof: a call graph execution profiler. Acm Sigplan Not. 39, 49–50.
Hennessy, J.L., Patterson, D.A., 2011. Computer Architecture, Fifth Edition: a Quantitative Approach. Morgan Kaufmann Publishers Inc.
Hill, M.D., Marty, M.R., 2008. Amdahl's law in the multicore era. Computer 41, 33–38.
Ki, S.J., Sugimura, T., Kim, A.S., 2015. OpenMP-accelerated SWAT simulation using Intel C and FORTRAN compilers: development and benchmark. Comput. Geosci. 75, 66–72.
Kiefer, T., Schlegel, B., Lehner, W., 2013. Experimental Evaluation of NUMA Effects on Database Management Systems, p. 360.
Li, T., Wang, G., Chen, J., Wang, H., 2011. Dynamic parallelization of hydrological model simulations. Environ. Model. Softw. 26, 1736–1746.
Li, X., Cheng, G.D., Liu, S.M., Xiao, Q., Ma, M.G., Jin, R., Che, T., Liu, Q.H., Wang, W.Z., Qi, Y., Wen, J.G., Li, H.Y., Zhu, G.F., Guo, J.W., Ran, Y.H., Wang, S.G., Zhu, Z.L., Zhou, J., Hu, X.L., Xu, Z.W., 2013. Heihe watershed allied telemetry experimental research (HiWATER): scientific objectives and experimental design. Bull. Am. Meteorol. Soc. 94, 1145–1160.
Liang, X., Guo, J.Z., Leung, L.R., 2004. Assessment of the effects of spatial resolutions on daily water flux simulations. J. Hydrol. 298, 287–310.
Liang, X., Wood, E.F., Lettenmaier, D.P., 1996. Surface soil moisture parameterization of the VIC-2L model: evaluation and modification. Glob. Planet. Change 13, 195–206.
Liu, J.Z., Zhu, A.X., Qin, C.Z., Wu, H., Jiang, J.C., 2016. A two-level parallelization method for distributed hydrological models. Environ. Model. Softw. 80, 175–184.
Neitsch, S.L., Arnold, J.G., Kiniry, J.R., Williams, J.R., 2011. Soil & Water Assessment Tool Theoretical Documentation Version 2009. Texas Water Resources Institute.
Pai, N., Saraswat, D., Srinivasan, R., 2012. Field_SWAT: a tool for mapping SWAT output to field boundaries. Comput. Geosci. 40, 175–184.
Palis, M.A., Liou, J.C., Wei, D.S.L., 1996. Task clustering and scheduling for distributed memory parallel architectures. Ieee Trans. Parallel Distrib. Syst. 7, 46–55.
Pan, X.D., Li, X., Cheng, G.D., Li, H.Y., He, X.B., 2015. Development and evaluation of a river-basin-scale high spatio-temporal precipitation data set using the WRF model: a case study of the Heihe River basin. Remote Sens. 7, 9230–9252.
Patterson, D.A., 2003. Comput. Archit. A Quant. Approach 12, 93.
Quinn, M.J., 2005. Parallel Program. C MPI OpenMP 5, 7.1–7.3.
Rajib, M.A., Merwade, V., Yu, Z.Q., 2016. Multi-objective calibration of a hydrologic model using spatially distributed remotely sensed/in-situ soil moisture. J. Hydrol. 536, 192–207.
Rathjens, H., Oppelt, N., 2012. SWATgrid: an interface for setting up SWAT in a grid-based discretization scheme. Comput. Geosci. 45, 161–167.
Rathjens, H., Oppelt, N., Bosch, D.D., Arnold, J.G., Volk, M., 2015. Development of a grid-based version of the SWAT landscape model. Hydrol. Process. 29, 900–914.
Rouholahnejad, E., Abbaspour, K.C., Vejdani, M., Srinivasan, R., Schulin, R., Lehmann, A., 2012. A parallelization framework for calibration of hydrological models. Environ. Model. Softw. 31, 28–36.
Saini, S., Jin, H., Hood, R., Barker, D., Mehrotra, P., Biswas, R., 2011. The impact of hyper-threading on processor resource utilization in production applications. In: International Conference on High PERFORMANCE Computing, pp. 1–10.
Vrugt, J.A., O Nuallain, B., Robinson, B.A., Bouten, W., Dekker, S.C., Sloot, P.M.A., 2006. Application of parallel computing to stochastic parameter estimation in environmental models. Comput. Geosci. 32, 1139–1155.
Wang, Y., Jung, Y., Supinie, T.A., Xue, M., 2013. A hybrid MPI-OpenMP parallel algorithm and performance analysis for an ensemble square root filter designed for multiscale observations. J. Atmos. Ocean. Technol. 30, 1382–1397.
Wu, Y.P., Li, T.J., Sun, L.Q., Chen, J., 2013. Parallelization of a hydrological model using the message passing interface. Environ. Model. Softw. 43, 124–132.
Xie, X.H., Zhang, D.X., 2010. Data assimilation for distributed hydrological catchment modeling via ensemble Kalman filter. Adv. Water Resour. 33, 678–690.
Yalew, S., van Griensven, A., Ray, N., Kokoszkiewicz, L., Betrie, G.D., 2013. Distributed computation of large scale SWAT models on the Grid. Environ. Model. Softw. 41, 223–230.
Zhang, L., Nan, Z.T., Yu, W.J., Ge, Y.C., 2015. Modeling land-use and land-cover change and hydrological responses under consistent climate change scenarios in the Heihe River basin, China. Water Resour. Manag. 29, 4701–4717.
Zhang, X., Izaurralde, R.C., Zong, Z., Zhao, K., Thomson, A.M., 2012. Evaluating the efficiency of a multi-core aware multi-objective optimization tool for calibrating the swat model. Trans. Asabe 55, 1723–1731.
Zhang, X.S., Beeson, P., Link, R., Manowitz, D., Izaurralde, R.C., Sadeghi, A., Thomson, A.M., Sahajpal, R., Srinivasan, R., Arnold, J.G., 2013. Efficient multi-objective calibration of a computationally intensive hydrologic model with parallel computing software in Python. Environ. Model. Softw. 46, 208–218.