J. Korea Water Resour. Assoc. Vol. 52, No. 11 (2019), pp. 887-894
doi: 10.3741/JKWRA.2019.52.11.887
pISSN 1226-6280
eISSN 2287-6138

Development of GPU-accelerated kinematic wave model using CUDA fortran

Kim, Boram[a] · Park, Seonryang[b] · Kim, Dae-Hong[c]*

[a]Graduate Student, Department of Civil Engineering, The University of Seoul, Seoul, Korea
[b]Graduate Student, Department of Civil Engineering, The University of Seoul, Seoul, Korea
[c]Professor, Department of Civil Engineering, The University of Seoul, Seoul, Korea

Paper number: 19-035
Received: 14 June 2019; Revised: 7 October 2019; Accepted: 7 October 2019

Abstract

We proposed a GPU (Grapic Processing Unit) accelerated kinematic wave model for rainfall runoff simulation and tested the accuracy and speed up performance of the proposed model. The governing equations are the kinematic wave equation for surface flow and the Green-Ampt model for infiltration. The kinematic wave equations were discretized using a finite volume method and CUDA fortran was used to implement the rainfall runoff model. Several numerical tests were conducted. The computed results of the GPU accelerated kinematic wave model were compared with several measured and other numerical results and reasonable agreements were observed from the comparisons. The speed up performance of the GPU accelerated model increased as the number of grids increased, achieving a maximum speed up of approximately 450 times compared to a CPU (Central Processing Unit) version, at least for the tested computing resources.

Keywords: GPU, Parallel computing, CUDA fortran, Kinematic wave model, Rainfall-runoff

CUDA fortran을 이용한 GPU 가속 운동파모형 개발

김보람[a] · 박선량[b] · 김대홍[c]*

[a]서울시립대학교 토목공학과 대학원생, [b]서울시립대학교 토목공학과 대학원생, [c]서울시립대학교 토목공학과 교수

요 지

분포형 강우유출모형의 수치모의 연산시간을 단축시키기 위해 GPU(Graphic Processing Unit)를 이용한 가속 운동파모형을 개발하고 정확성과 연산속도에 대한 성능을 검토하였다. 분포형모형의 지배방정식은 운동파모형과 Green-Ampt모형으로 구성되었고, 운동파모형은 유한체적법을 이용하여 이산화 하였다. GPU 가속 운동파모형 개발을 위해 CUDA fortran을 이용하였다. 개발된 모형을 이용하여 이상적인 유역에서 발생하는 강우유출현상을 모의 하였고, 다른 모형 및 실험결과와의 비교를 통하여 개발된 GPU 가속 운동파모형이 비교적 정확하게 유출량을 계산할 수 있음을 확인하였다. 동일한 유한체적법을 이용한 CPU(Central Processing Unit) 기반의 강우유출모형과 비교할 경우, GPU 가속모형의 연산시간 단축비율은 격자의 수가 증가할수록 높아졌으며, 본 연구에 사용된 장비를 기준으로 최대 450배 정도 단축됨을 확인하였다.

핵심용어: 그래픽처리장치, 병렬 컴퓨팅, 쿠다 포트란, 운동파모형, 강우유출

1. 서 론

우리나라에서 발생하는 대부분의 자연재해는 여름철 집중호우와 태풍 등으로 인한 수재해이다(Kim *et al*., 2013a). 최근 들어, 집중호우로 인한 범람과 침수 등의 현상이 증가하고 있는 추세이다. 이와 같은 수재해를 예측하고 방지하기 위한 대책 중 하나로써 강우유출모형에 대한 연구가 지속적으로 활발하게 수행되고 있다(Esteves *et al*., 2000; Tayfur and Kavvas, 1994; Rousseau *et al*., 2015). 특히 컴퓨터의 연산성능 향상으로 인해 강우와 지형 및 토양특성의 공간분포를 고려하여 강우유출을 계산할 수 있는 물리기반의 분포형 강우유출모형에 대한 활용이 증가하고 있다(Park *et al*., 2007). 그러나 분포형

***Corresponding Author.** Tel: +82-2-6490-2432
E-mail: dhkimhyd@uos.ac.kr (D.-H. Kim)

모형은 집중형모형에 비해 입력자료의 구축에 많은 시간과 노력이 필요하며, 특히 유출 계산 시 컴퓨터의 연산 수행시간이 상대적으로 오래 걸리는 단점이 있다. 이와 같은 이유로 인하여 분포형모형을 대유역에 적용하는 경우 격자의 해상도를 충분히 높이지 않고 사용되어 왔다(Chung *et al*., 2010).

지속적인 컴퓨터의 연산성능 향상으로 인하여, 전산유체역학 분야에서는 유체흐름을 보다 사실적으로 계산하고 표현하려는 연구가 활발하게 수행되고 있다. 수치해석을 이용하여 이와 같은 목적을 달성하기 위해서는 고해상도로 계산영역을 분할해야 하므로, 연산수행 시간을 단축시키기 위한 연구가 매우 활발하게 진행되어왔다. 기존에는 여러 개의 중앙처리장치(Central Processing Unit, CPU)를 이용하는 MPI 기법이나 다수의 Thread를 이용하는 OpenMP 기법과 같이, CPU 기반의 병렬처리 기법이 활발하게 연구 및 활용되어왔다. 최근에는 그래픽처리장치(Graphic Processing Unit, GPU)를 이용한 병렬처리기법에 관한 연구가 활발히 진행 중이다.

CPU 내부에는 다양한 작업을 수행하기 위해 물리적 구조가 복잡한 산술논리장치(Arithmetic Logic Unit, ALU)가 있다. 반면 GPU 내부에는 비교적 단순한 ALU가 있지만 CPU보다 다수의 ALU로 구성되어 있어서, CPU보다 한 번에 더 많은 연산수행이 가능하다. GPU는 CPU와 하드웨어적 차이로 인해 CPU와는 다른 방식의 프로그래밍 기법이 필요하다. GPU를 이용한 프로그래밍은 NVIDIA사의 CUDA (Compute Unified Device Architecture)와 Khronos Group의 OpenCL (Open Computing Language)이 대표적이다(Vanderbauwhede and Takemi, 2013). 이들 중 GPU 제조업체인 NVIDIA는 컴파일러(compiler) 개발사인 The Portland Group과 공동으로 GPU를 연산에 이용할 수 있는 컴파일러를 개발했으며, 그중 하나가 CUDA fortran 컴파일러이다. 지난 수년에 걸쳐 전산유체역학 분야에서는 GPU를 이용한 병렬연산에 대한 연구가 진행되어왔다. Gomez Gesteira *et al*. (2012a; 2012b)은 자유표면 유체해석이 가능한 GPU 기반 smoothed particle hydrodynamics 모형인 SPHysics (www.sphysics.org)를 개발하였다. Vanderbauwhede and Takemi (2013)는 OpenCL 기반 Weather Research and Forecasting (WRF) 모형을 개발했으며, GPU를 이용한 모형의 연산 수행시간은 CPU를 이용한 모형의 연산 수행시간에 비해 $O(10^2)$배 단축되었다. Kim *et al*. (2013b)은 CUDA fortran 기반 WRF 모형을 개발하였으며, GPU를 이용한 연산 수행시간이 CPU를 이용한 연산 수행시간에 비해 약 5배 단축되었다. Chang *et al*. (2014)은 고차정확도 수치기법기반의 압축성 유동 해석모형 개발에 CUDA를 이용하였다. GPU를 이용한 연산수행시간은 CPU의 12개 코어를 사용한 OpenMP 병렬연산 수행시간보다 최대 약 2배 단축되었다.

본 연구에서는 GPU 가속기법을 이용하여 강우유출을 물리적으로 해석하는데 활발하게 연구되고 있는 운동파모형을 개발하였다. 또한 개발된 GPU 가속 운동파모형의 정확성과 연산수행시간 성능에 대한 검토 결과를 제시하였다.

2. 지배방정식

2.1 운동파모형

지표유출해석을 위한 운동파모형의 연속방정식과 운동방정식은 Eqs. (1) and (2)와 같다(Liu *et al*., 2004).

$$\frac{\partial h}{\partial t}+\frac{\partial q_x}{\partial x}+\frac{\partial q_y}{\partial y}=r-f \tag{1}$$

$$S_{0x}=S_{fx} \tag{2a}$$

$$S_{0y}=S_{fy} \tag{2b}$$

여기서, (x,y)는 평면공간좌표의 방향을 나타내고, h는 수심, t는 시간이다. q_x와 q_y는 x와 y축 방향 단위폭 당 유량이다. r은 강우강도, f는 침투율이다. S_{0x}와 S_{0y}는 x와 y축 방향 바닥경사이며, S_{fx}와 S_{fy}는 x와 y축 방향 마찰경사이다. Eqs. (2a) and (2b)의 마찰경사는 조도계수를 이용하여 Eqs. (3a) and (3b)로 나타낼 수 있다(Govindaraju *et al*., 1992).

$$S_{fx}=n^2\frac{u\sqrt{u^2+v^2}}{h^{4/3}} \tag{3a}$$

$$S_{fy}=n^2\frac{v\sqrt{u^2+v^2}}{h^{4/3}} \tag{3b}$$

여기서, n은 조도계수, u와 v는 x와 y축 방향 유속이다. Eqs. (2) and (3)을 정리하여 단위폭 당 유량으로 나타내면 Eqs. (4a) and (4b)와 같이 정리할 수 있다.

$$q_x=-sign(S_{0x})\frac{1}{n}h^{5/3}\frac{S_{0x}^{1/2}}{\left\{1+\left(\frac{S_{0y}}{S_{0x}}\right)^2\right\}^{1/4}} \tag{4a}$$

$$q_y=-sign(S_{0y})\frac{1}{n}h^{5/3}\frac{S_{0y}^{1/2}}{\left\{1+\left(\frac{S_{0x}}{S_{0y}}\right)^2\right\}^{1/4}} \tag{4b}$$

Eq. (1)을 유한체적법으로 이산화하면 Eq. (5)와 같다(Liu *et al.*, 2004).

$$h_{i,j}^{k+1} = h_{i,j}^{k} - \frac{\Delta t}{\Delta x \Delta y}(q_{x_{i+1/2,j}} \Delta y_{i+1/2,j} - q_{x_{i-1/2,j}} \Delta y_{i-1/2,j} + q_{y_{i,j+1/2}} \Delta x_{i,j+1/2} - q_{y_{i,j-1/2}} \Delta x_{i,j-1/2}) + r_{i,j}^{k+1} \Delta t - f_{i,j}^{k+1} \Delta t \quad (5)$$

여기서, (i,j)는 이산화된 공간격자의 순서를 나타낸다. Δt는 시간간격, k는 시간단계, Δx와 Δy는 x축과 y축 방향 격자크기이다. Δt 동안의 x와 y축 방향 유입량은 Eqs. (6) and (7)과 같다.

$$q_{x_{i-1/2,j}} \Delta y_{i-1/2,j} \Delta t = [\max(q_{x_{i-1,j}}, 0) \Delta y_{i-1/2,j} - \min(q_{x_{i+1,j}}, 0) \Delta y_{i+1/2,j}] \Delta t \quad (6)$$

$$q_{y_{i,j-1/2}} \Delta x_{i,j-1/2} \Delta t = [\max(q_{y_{i,j-1}}, 0) \Delta x_{i,j-1/2} - \min(q_{x_{i,j+1}}, 0) \Delta x_{i,j+1/2}] \Delta t \quad (7)$$

Δt동안의 x축과 y축 방향 유출량은 Eqs. (8) and (9)와 같으며, 수치해석기법에 대한 보다 자세한 내용은 Liu *et al.* (2004)에 기술되어있다.

$$q_{x_{i+1/2,j}} \Delta y_{i+1/2,j} \Delta t = q_{x_{i,j}} \begin{cases} \Delta y_{i+1/2,j} \Delta t & , q_{x_{i,j}} \geq 0 \\ \Delta y_{i-1/2,j} \Delta t & , q_{x_{i,j}} < 0 \end{cases} \quad (8)$$

$$q_{y_{i,j+1/2}} \Delta x_{i,j+1/2} \Delta t = q_{y_{i,j}} \begin{cases} \Delta x_{i,j+1/2} \Delta t & , q_{y_{i,j}} \geq 0 \\ \Delta x_{i,j-1/2} \Delta t & , q_{y_{i,j}} < 0 \end{cases} \quad (9)$$

2.2 침투 모형

균질한 흙을 통한 연직방향 침투량을 산정하기 위한 다수의 모형들이 제안되었다(Green and Ampt, 1911; Horton, 1939; Mein and Larson, 1973). 실제 침투는 흙의 함수특성, 균질성, 포화도 등에 영향을 받는다. 따라서 본 연구에서는 다양한 침투모형 중에서 물리기반의 침투모형인 Green-Ampt 모형을 이용하였다. Green-Ampt모형은 1차원 연직흐름, 침윤전선에서의 일정한 모관흡수력, 그리고 침윤전선 위 부분에서의 함수량 결손과 투수계수가 일정하다는 가정을 하고 있다. Mein and Larson (1973)은 Green-Ampt모형을 일부 수정하여 담수시간에 따른 강우강도에 대한 침투율을 Eq. (10)과 같이 제시하였다.

$$\begin{cases} f = r & t \leq t_p \\ f = K_s\left(1 + \frac{\Delta h_f}{F}\right) & t > t_p \end{cases} \quad (10)$$

여기서, 담수시간은 $t_p = F_p / r$이고, $F_p = h_f \Delta\theta / (r/K_s - 1)$, K_s는 투수계수, h_f는 습윤전선의 모관흡인수두이다. Eq. (10)의 $t > t_p$에 대한 Green-Ampt모형은 초기 포화조건에서 적용되므로 누가침투량은 다음과 같이 유도된다.

$$F(t) = K_s[t - (t_p - t_s)] + h_f \Delta\theta \ln\left(1 + \frac{F(t)}{h_f \Delta\theta}\right) \quad (11)$$

여기서, t_s는 $t = 0$부터 침투율과 강우강도가 같아질 때까지의 시간이고 $\Delta\theta$는 토양수분부족량이다. Eq. (11)의 비선형항의 해를 구하기 위해 Newton-Raphson기법을 이용하였다.

3. GPU 가속 기법

3.1 GPU의 하드웨어 구조와 특징

GPU는 처리할 수 있는 인스트럭션(instruction)이 단순한 산술연산에 국한되어 있어 CPU에 비해 단순한 제어부와 훨씬 작은 크기의 캐시 메모리를 가지고 있다. 하지만, GPU는 수천 개의 스트림 프로세싱(stream processing)을 통해 동일한 형태의 산술연산을 동시다발적으로 처리하는데 매우 우수한 장점이 있다(Ruetsch and Fatica, 2013). Fig. 1은 CPU와 GPU의 하드웨어 구조를 간략하게 나타내고 있다. 제어장치(Control unit)는 프로세서의 조작을 지시하는 장치로 입출력 장치 간 통신 및 조율을 제어한다. ALU는 산술연산과 논리연산을 계산하는 디지털 회로이다. Fig. 1과 같이 CPU는 소수의 ALU로 구성되어있는 반면, GPU는 다수의 ALU를 가지고 있다. 본 연구에 사용된 장비를 기준으로 CPU (Intel Xeon

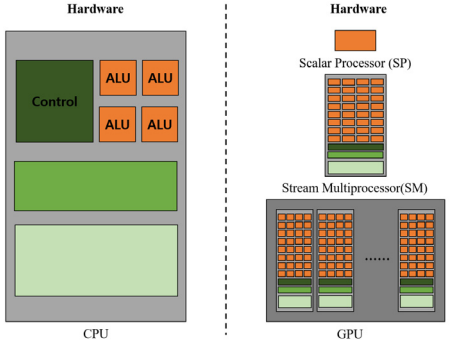

Fig. 1. Hardware architecture of CPU and GPU (Kim, 2019)

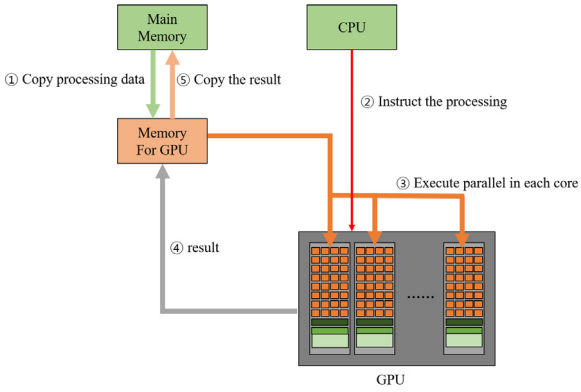

Fig. 2. CUDA data processing flow (Kim, 2019)

```
1  do i = 1, nx
2      do j = 1,ny
3          A(i,j) = B(i,j)+C(i,j)
4      enddo
5  Enddo
```

(a) CPU

```
1  i = threadidx%x + (blockidx%x-1) * blockDim%x
2  j = threadidx%y + (blockIdx%y-1) * blockDim%y
3
4  A(i,j) = B(i,j)+C(i,j)
5
```

(b) GPU

Fig. 3. Example of GPU programming

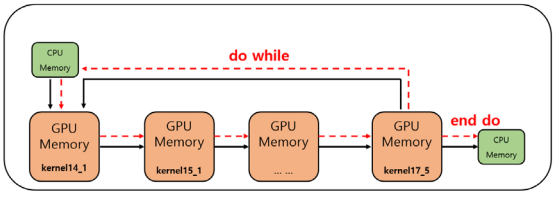

Fig. 4. Structure of gpu-accelerated kinematic wave model. Solid line: data transfer. Dashed line: flow sequence

```
1  do while (ts.le.t end)
2      ts = ts+dt
3      dev_ts = ts
4      call kernel14_1<<<gridDim,blockDim>>>(dev_h,dev_ts)
5      call kernel15_1<<<gridDim,blockDim>>>(dev_h,dev_hx)
6      ... ...
7      call kernel17_4<<<gridDim,blockDim>>>(dev_hn,dev_h,dev_qx,dev_qy)
8      call kernel17_5<<<gridDim,blockDim>>>(dev_hn,dev_h,dev_qx,dev_qy)
9  enddo
10 qy = dev_qy
```

Fig. 5. GPU-accelerated code for kinematic wave model

E5-2620 v2 @ 2.10GHz)에는 ALU를 포함하고 있는 코어가 6개이며, GPU (NVIDIA GeForce GTX TITAN Z)에는 5760개의 코어가 있다.

Fig. 2는 CUDA의 데이터 처리 흐름을 나타낸다. CPU와 GPU는 자신이 제어할 수 있는 독립적인 메모리를 가지고 있으며, 상대방의 메모리는 제어할 수 없다. 따라서 먼저 CPU 메모리에서 GPU 메모리로 데이터를 전송하고, CPU가 GPU에 데이터 처리를 명령한다. 데이터 처리 명령을 받은 GPU는 각 코어에서 데이터를 병렬로 처리하고, 결과를 GPU 메모리를 거쳐 CPU 메모리로 전송시킨다. GPU를 사용한 데이터 처리 흐름은 CPU와 GPU의 하드웨어 구조의 차이로 인해 CPU만을 이용한 데이터 처리 흐름과 비교 시 CPU와 GPU간의 데이터 전송이라는 추가적인 수행 과정이 발생하게 된다. 따라서 이와 같은 CPU와 GPU간의 데이터 전송으로 인한 추가 소요시간이 발생한다.

3.2 GPU 가속 프로그래밍

Fig. 3은 CPU 기반의 순차연산기법과 GPU 기반의 병렬연산기법을 활용하여 2개 배열의 합을 구하는 코드의 한 가지 예이다(Kim, 2019). 본 연구에서 GPU 가속모형의 개발을 위해 사용한 CUDA는 계산영역내의 격자 개수에 대한 그리드(Grid)와 블록(Block)을 생성한다. 그리드는 블록의 모임이고, 블록은 쓰레드(thread)의 모임이며, 쓰레드는 GPU 내에서 작동되는 하나의 코어를 의미한다. CUDA의 병렬연산은 코어 하나에 한 개의 명령어가 할당되고, 한 블록 내의 쓰레드들이 동시에 병렬로 연산되는 방식이다. GPU 프로그램에서는 Fig. 3(b)와 같이 i와 j를 사용한 반복문으로 표현하지 않으며, 필요시 반복문을 표현 할 수도 있다. Fig. 3과 같이 쓰레드의 배열은 blockDim 변수에 저장되고, threadidx 변수는 index가 주어진다. 또한 블록의 배열은 gridDim 변수에 저장되고, blockidx 변수는 index를 주어진다(NVIDIA, 2011).

연산 수행시간을 단축시키기 위해서 GPU 가속 수치모형 구현 시 CPU와 GPU간 대용량 데이터 전송이 빈번하게 발생하지 않도록 프로그래밍을 해야 한다. 따라서 대용량 연산시에는 CPU와 GPU간의 자료전송 빈도를 줄이고 GPU에서 연속적으로 연산이 수행되도록 프로그래밍을 해야 한다. Figs. 4 and 5는 본 연구에서 사용된 GPU 가속 운동파모형의 프로그램체계를 간략히 나타낸 것으로, 초기조건의 입력과 매우 단순한 연산을 제외한 주요부분이 GPU내에서 연산이 연속적으로 수행됨을 나타낸다. 예를 들어, 해당 코드의 일부를 보여주는 Fig. 5의 3행은 시간(t)을 GPU로 전송하는 과정을 나타내는데, 이는 하나의 변수로써 처리시간이 매우 짧아 수치모

의의 전체시간에 영향을 거의 주지 않는다. Fig. 5의 4~8행은 2장에 제시되어 있는 운동모형의 이산화식과 침투모형을 풀기 위한 부분으로 대용량의 데이터와 연산과정이 필요하여 GPU 내에서 연속적으로 수행되도록 하였다.

4. 수치모의결과

4.1 GPU 가속모형 모의결과 검토

4.1.1 Delestre *et al.* (2008)의 수리실험

Delestre *et al.* (2008)의 수리실험 결과와 GPU 가속 운동파모형을 이용한 수치모의 결과를 비교하여 제시된 모형의 정확성을 검토하였다. Fig. 6은 수리실험의 측면도로써, 바닥의 길이는 $L_x = 4\,m$이고, 바닥경사는 $S_{0x} = 0.0496$이다. 강우강도는 하류단으로부터 0.05 m까지를 제외한 전 영역에 모의 시작 후 5초부터 125초까지 $r = 50.76\,mm/hr$의 강우조건을 부여하였다. 수치모의에 사용된 공간격자의 크기는 $\Delta x = \Delta y = 0.01\ m$, 시간간격은 $\Delta t = 0.01\ s$이고 조도계수는 $n = 0.013\ s/m^{1/3}$을 이용하였다.

Fig. 7은 수로 하류단에서의 유출수문곡선으로써, GPU 가속 운동파모형의 계산결과는 Delestre *et al.* (2008)의 실험결과와 잘 일치하고 있다. 또한 GPU 가속 운동파모형과 동일한 수치기법을 이용하여 개발된 CPU 기반 운동파모형의 수치모의 결과와도 매우 유사하다.

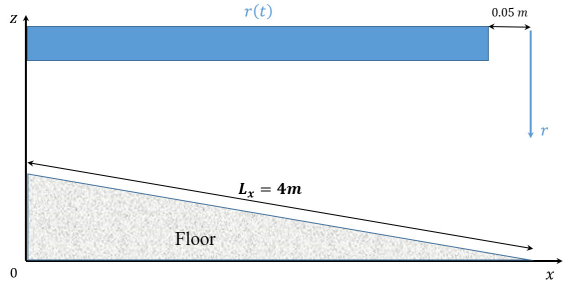

Fig. 6. Schematic diagram of rainfall runoff experiment (Delestre *et al.*, 2008)

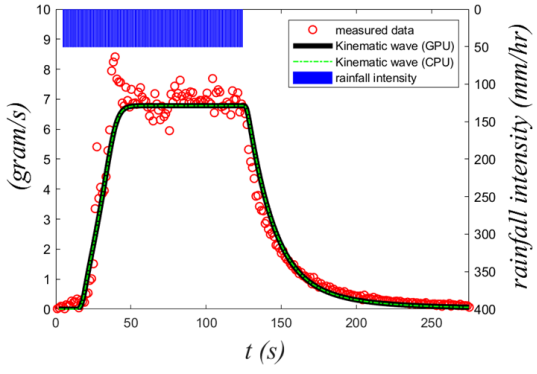

Fig. 7. Measured and computed runoff hydrographs at outlet

4.1.2 Fernández Pato *et al.* (2016)의 수치모의

침투가 발생하는 일차원 유역에서의 검증을 수행하기 위하여, Fernández Pato *et al.* (2016)이 제시한 천수방정식(Shallow water equation)모형의 수치모의 결과와 본 연구에서 제시한 GPU 가속 운동파모형의 수치모의 결과를 비교하였다. 유역의 길이는 2000 m이고 바닥경사는 전구간에 걸쳐 $S_0 = 0.005$이다. 유역의 전구간에 걸쳐 $r = 450\ m/hr$의 강우가 125분 동안 지속되며, 침투관련 매개변수는 $K_s = 3.272\times10^{-5}\ m/s$, $h_f = 0.0495\ m$, $\Delta\theta = 0.38$이고, 조도계수는 $n = 0.03\ s/m^{1/3}$이다. 격자크기는 $\Delta x = \Delta y = 1\ m$, 시간간격은 $\Delta t = 0.1\ s$를 이용하였다. Fig. 8은 유역출구에서의 유출수문곡선으로 GPU 가속 운동파모형의 모의결과가 CPU 기반 운동파모형의 모의결과 및 Fernández Pato *et al.* (2016)이 제시한 천수방정식을 이용한 수치모의 결과와 잘 일치하고 있음을 보여준다.

4.1.3 V-형 유역에 대한 수치모의

2차원 유역에서의 검증을 위하여, 본 연구에서 개발된 GPU 가속 운동파모형을 V-형태의 지형 (Fig. 9)에 적용하고

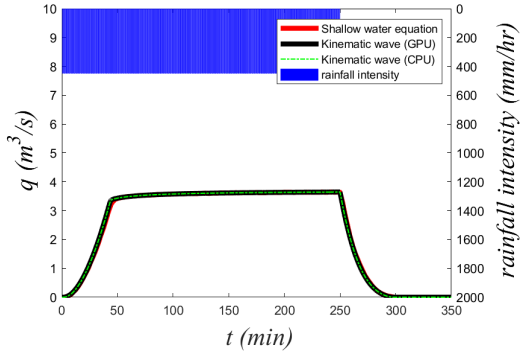

Fig. 8. Computed results of kinematic wave model and shallow water equation model

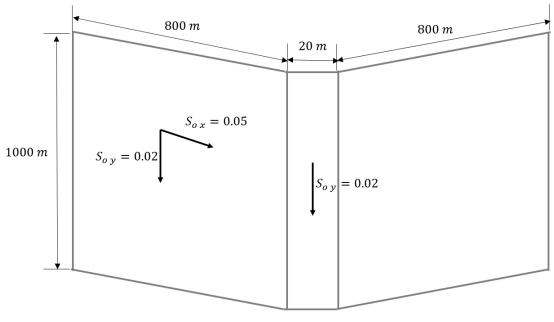

Fig. 9. Geometry of V-shaped catchment

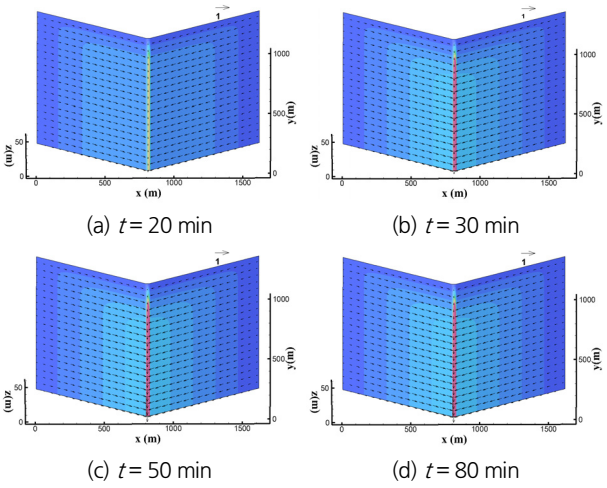

Fig. 10. Computed results of GPU-accelerated kinematic wave model. Vector: velocity. Colormap: discharge

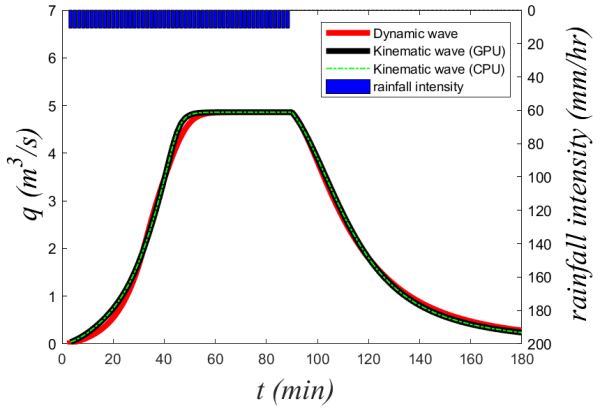

Fig. 11. Computed hydrographs of kinematic wave and dynamic wave models

Di Giammarco *et al.* (1996)이 제시한 동역학파모형의 수치모의 결과와 비교하였다. Fig. 10은 V-형 유역으로, 중앙수로는 20 m × 1000 m의 평면이고 좌우측 유역은 800 m × 1000 m인 평면으로 이루어졌다. 바닥경사는 중앙수로와 수직방향으로는 $S_{0x} = 0.05$이고, 수로의 종방향으로는 $S_{0y} = 0.02$이다. 강우강도는 전 영역에서 $r = 10.8\ mm/hr$이다. 수치모의 시 격자크기는 $\Delta x = \Delta y = 10\ m$, 시간간격은 $\Delta t = 1.0\ s$이다. 조도계수는 중앙수로에서 $n = 0.15\ s/m^{1/3}$이고, 좌우 유역에서 $n = 0.015\ s/m^{1/3}$이다.

Fig. 10은 시간에 따른 GPU 가속 운동파모형 수치모의 결과로 유량과 유속을 나타낸다. Fig. 11은 유역 하류단에서의 유출수문곡선으로 GPU 가속 운동파모형, CPU 기반 운동파모형 및 동역학파모형(Di Giammarco *et al.*, 1996)의 수치모의 결과이다. Fig. 11과 Figs. 10(a)~10(c)와 같이 강우가 시작

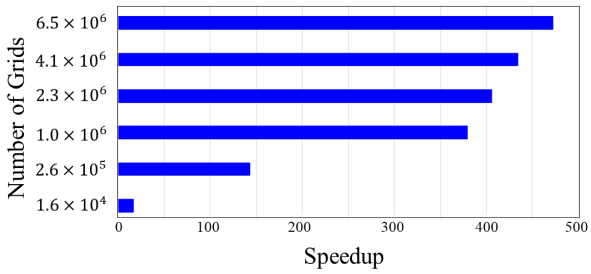

Fig. 12. Speedup of GPU-accelerated kinematic wave model

된 후 t = 20 min~t = 50 min 동안 중앙 수로로 물이 모이고 하류단으로 이동하여, 하류단에서의 유량이 증가하고 있음을 나타내고 있다. 또한 t = 50 min~t = 90 min에는 수로의 하류부에서 유량이 일정하게 유지되고 있으며, 이는 Fig. 10(d)가 유량이 일정하게 유지되고 있을 때의 유량과 유속을 나타낸다. Fig. 11에서 관찰 할 수 있듯이 GPU 가속 운동파모형의 계산결과는 동역학파모형의 결과 및 CPU 기반 운동파모형의 결과와 잘 일치하고 있다.

4.2 GPU 가속 운동파모형의 연산속도

GPU 가속 운동파모형의 연산 수행시간에 대한 성능을 조사하기 위해 GPU 가속 운동파모형의 전체 프로그램 수행시간과 CPU 기반 운동파모형의 전체 프로그램 수행시간을 측정하여 비교하였다. GPU 가속 운동파모형의 연산 수행시간에 대한 성능은 속도향상의 척도인 Speedup을 이용하여 계산하였다.

$$\text{Speedup} = \frac{\text{CPUtime}}{\text{GPUtime}} \tag{12}$$

여기서, GPU time은 GPU 가속 수치모형의 전체 프로그램 수행시간이며, GPU time은 CPU 기반 수치모형의 전체 프로그램 수행시간이다. GPU 가속 운동파모형의 수치모의 시 동일한 예제에서 격자 개수를 다양하게 구성하여 격자 개수와 Speedup의 관계를 확인하였다. Fig. 12는 Fig. 9의 V-shaped 유역에 대한 GPU 가속 운동파모형의 Speedup 결과로써, 격자 개수를 1.6×10^4부터 6.5×10^6까지 증가시킬 때, Speedup도 증가하고 있음을 확인하였다. GPU 가속 운동파모형은 CPU 기반 운동파모형보다 최대 450배 정도 Speedup 되었으며, 이는 대규모의 연산 수행에 있어 GPU를 이용하는 것이 1개의 CPU만을 이용하는 것보다 연산 수행시간을 단축시킬 수 있음을 나타낸다. 본 연구에서는 운동파모형 수행시간에 대한 Speedup 결과 중 하나의 결과만을 제시하였지만, 다른 유역을

대상으로 한 모의시에도 대략적으로 비슷한 Speedup 결과가 도출되었다.

5. 결 론

본 연구에서는 강우유출모의를 위한 운동파모형의 연산 수행시간을 단축시키기 위해 CUDA fortran 컴파일러를 사용한 GPU 가속 운동파모형을 개발하고 이상적인 유역을 대상으로 수치모의 한 유출수문곡선의 정확성을 검증하며 연산 수행시간에 대한 성능을 조사하였다. 본 연구에서 제시한 GPU 가속 운동파모형을 이용한 수치모의 결과는 선행된 실험 및 수치모의 연구 결과와 잘 일치하고 있다. GPU 가속 운동파모형은 격자의 개수가 증가 할수록 Speedup이 최대 약 450배까지 증가하였다. 그러므로 대용량 연산수행에 있어 GPU 가속 운동파모형은 기존 CPU 기반 운동파모형에 비해 연산 수행시간에 대한 성능이 높으며, 지속적인 개발을 수행할 경우에는 실제 대규모 유역에 적용 가능할 것으로 판단된다.

본 연구에서 제시된 결과는 GPU 가속 운동파모형의 정확성과 연산 수행시간에 대한 성능을 검토하기 위한 초기단계의 성과이다. 따라서 GPU 가속 운동파모형의 정확성과 적용성에 대한 평가를 위해서는 실제유역을 포함한 보다 다양한 지형에 대한 적용결과의 평가가 필요하다. 또한 TVD 기법과 같은 보다 수치적으로 안정적인 기법을 이용하여 GPU 가속 운동파모형을 개발을 시도한다면, 보다 안정적이고 적용성이 높은 분포형 강우유출모형의 개발이 가능할 것으로 판단된다.

감사의 글

이 논문은 2018년도 서울시립대학교 교내학술연구비에 의하여 지원되었음.

References

Chang, T. K., Park, J. S., and Kim, C. (2014). *Efficient computation of compressible flow by higher-order method accelerated using GPU*. M. d. dissertation, Seoul National University.

Chung, S. Y., Park, J. H., Hur, Y. T., and Jung, K. S., (2010). "Application of Mpi technique for distributed rainfall-runoff model." *Journal of Korea Water Resources Association*, KWRA, Vol. 43, No. 8, pp. 747-755.

Delestre, O., Cordier, S., James, F., and Darboux, F. (2008). "Simulation of rainwater overland-flow." *In: 12th International Conference on Hyperbolic Problems*, *American Mathematical Society*, Vol. 67, Maryland, USA, pp. 537-546.

Di Giammarco, P., Todini, E., and Lamberti, P. (1996). "A conservative finite elements approach to overland flow: the control volume finite element formulation." *Journal of Hydrology*, Vol. 175, No. 1-4, pp. 267-291.

Esteves, M., Faucher, X., Galle, S., and Vauclin, M. (2000). "Overland flow and infiltration modelling for small plots during unsteady rain: Numerical results versus observed values." *Journal of Hydrology*, Vol. 228, No. 3-4, pp. 265-282.

Fernández Pato, J., Caviedes-Voulliéme, D., and García-Navarro, P. (2016). "Rainfall/runoff simulation with 2d full shallow water equations: Sensitivity analysis and calibration of infiltration parameters." *Journal of Hydrology*, Vol. 536, pp. 496-513.

Gomez Gesteira, M., Crespo, A. J., Rogers, B. D., Dalrymple, R. A., Dominguez, J. M., and Barreiro, A., (2012a). "Sphysics-development of a freesurface fluid solver-part 2: Efficiency and test cases." *Computers & Geosciences*, Vol. 48, pp. 300-307.

Gomez Gesteira, M., Rogers, B. D., Crespo, A. J., Dalrymple, R. A., Narayanaswamy, M., and Dominguez, J. M., (2012b). "Sphysics-development of a free-surface fluid solver-part 1: Theory and formulations." *Computers & Geosciences*, Vol. 48, pp. 289-299.

Govindaraju, R. S., Kavvas, M. L., and Tayfur, G. (1992). "A simplified model for two-dimensional overland flows." *Advances in Water Resources*, Vol. 15, No. 2, pp. 133-141.

Green, W. H., and Ampt, G. (1911). "Studies on soil phyics." *The Journal of Agricultural Science*, Vol. 4, No. 1, pp. 1-24.

Horton, R. E. (1939). "Analysis of runoff-plat experiments with varying infiltration-capacity." *Eos, Transactions American Geophysical Union*, Vol. 20, No. 4, pp. 693-711.

Kim, B. R. (2019). *Development of GPU-accelerated numerical model for surface and ground water flow*. Ph. D. dissertation, University of Seoul.

Kim, S. W., Jung, S. J., Choi, E. K., Kim, S. H., Lee, K. H., and Park, D. G. (2013a). "An Analysis of the current status of disasters occurring on the steep slopes in Korea." *Journal of Environmental Science International*, Vol. 22, No. 11, pp. 1529-1538.

Kim, Y. T., Lee, Y. L., and Chung, K. Y. (2013b). "WRF physics models using GP-GPUs with CUDA fortran." *Korean Meteorological Society*, Vol. 23, No. 2, pp. 231-235.

Liu, Q., Chen, L., Li, J., and Singh, V. (2004). "Two-dimensional kinematic wave model of overland-flow." *Journal of Hydrology*, Vol. 291, No. 1-2, pp. 28-41.

Mein, R. G., and Larson, C. L. (1973). "Modeling infiltration during a steady rain." *Water Resources Research*, Vol. 9, No. 2, pp. 384-394.

NVIDIA (2011). *Cuda c programming guide version 4.0*. NVIDIA Corporation 4. USA.

Park, J. H., Kang, B. S., Lee, G. S., and Lee, E. R. (2007). "Flood runoff analysis using radar rainfall and vflo model for Namgang Dam watershed." *Journal of the Korean Association of Geographic Information Studies*, Vol. 10, No. 3, pp. 13-21.

Rousseau, M., Cerdan, O., Delestre, O., Dupros, F., James, F., and Cordier, S. (2015). "Overland flow modeling with the shallow water equations using a well-balanced numerical scheme: Better predictions or just more complexity." *Journal of Hydrologic Engineering*, Vol. 20, No. 10, p. 04015012.

Ruetsch, G., and Fatica, M. (2013). *CUDA Fortran for scientists and engineers: best practices for efficient CUDA Fortran programming*. Elsevier, Amsterdam.

Tayfur, G., and Kavvas, M. L. (1994). "Spatially averaged conservation equations for interacting rill-interrill area overland flows." *Journal of Hydraulic Engineering*, Vol. 120, No. 12, pp. 1426-1448.

Vanderbauwhede, W., and Takemi, T., (2013). "An investigation into the feasibility and benefits of gpu/multicore acceleration of the weather research and forecasting model." *In: 2013 International Conference on High Performance Computing & Simulation* (HPCS), IEEE, Helsinki, Finland, pp. 482-489.