Accelerating Google's Flood Forecasting Initiative with Tensor Processing Units

Damien Pierce
dmpierce@google.com
Sept 2020

with V. Anisimov, A. Boral, Y. Chen, L. Hu, S. Nevo, Y. Shafi
Google Research
Flood forecasting

- Affects hundreds of millions of people
- Thousands of fatalities per year
- Flood forecasting is an effective mitigation tool
  - Can reduce fatalities and economic impacts by a third*

Flood forecasting ingredients

- Real-time and forecasted water level measurements
- High resolution Digital Elevation Models (DEM)
- Forecasting techniques: some combination of
  - Hydrological modeling
  - **Hydraulic modeling**
  - Machine learning
Hydraulic modeling: 2D shallow water equations

\[
\frac{\partial h}{\partial t} + \frac{\partial q_x}{\partial x} + \frac{\partial q_y}{\partial y} = 0
\]

\[
\frac{\partial q_i}{\partial t} + gh \frac{\partial (h + z)}{\partial i} + \frac{gn^2}{h^{7/3}} \|\mathbf{q}\| q_i = 0, \quad i \in \{x, y\}
\]

\(q = \text{flux} \) [discharge per unit width, \(L^2/T\)]

\(h = \text{water height}\)

\(z = \text{surface elevation}\)

\(n = \text{Manning friction coefficient}\)
USGS 3DEP Map
As of Aug 2018

https://nationalmap.gov/preview/3DEP
Arkansas River

Flooded in May, 2019

Region modeled:

990 sq km
244k acres

aspect ratio: 2.15 : 1

Almost 1 billion square meters → 1 billion grid points in 1m simulation
Hydraulic Model Simulation

- Main parameter is the discharge at the input boundary (volume of water per unit time)
- Run to (close to) steady state (2 days)
- Run with various discharges
- Results compared to satellite images
- $\rightarrow$ Discharge = $15k \text{ m}^3/\text{s}$
Table 3. Streamgage information related to calculation of annual exceedance probability for the May to June 2019 flood event along Arkansas.

[AEP, annual exceedance probability; USGS, U.S. Geological Survey; ft, foot; ft³/s, cubic foot per second]

<table>
<thead>
<tr>
<th>USGS streamgage number¹</th>
<th>USGS streamgage name</th>
<th>Date of peak streamflow</th>
<th>Peak streamflow for May to June 2019 flood</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Peak gage height (ft)</td>
</tr>
<tr>
<td>07152500</td>
<td>Arkansas River at Ralston, Okla.</td>
<td>5/23/2019</td>
<td>22.14</td>
</tr>
<tr>
<td>07164500</td>
<td>Arkansas River at Tulsa, Okla.</td>
<td>5/29/2019</td>
<td>23.51</td>
</tr>
<tr>
<td>07165570</td>
<td>Arkansas River near Haskell, Okla.</td>
<td>5/29/2019</td>
<td>24.24</td>
</tr>
<tr>
<td>07194500</td>
<td>Arkansas River near Muskogee, Okla.²</td>
<td>5/26/2019</td>
<td>46.39</td>
</tr>
<tr>
<td>07250550</td>
<td>Arkansas River at James W. Trimble L&amp;D near Van Buren, Ark.³</td>
<td>5/31/2019</td>
<td>406.96</td>
</tr>
<tr>
<td>07258000</td>
<td>Arkansas River at Dardanelle, Ark.³</td>
<td>5/30/2019</td>
<td><strong>45.91</strong></td>
</tr>
<tr>
<td>07263450</td>
<td>Arkansas River at Murray Dam near Little Rock, Ark.³</td>
<td>6/4/2019</td>
<td>259.75</td>
</tr>
</tbody>
</table>

565k ft³ / s = 16k m³ / s
15k discharge simulation overlay
Observations about sim result

- USGS 3DEP Lidar provides an excellent DEM
  - captures bare earth beneath trees
  - includes bathymetry
- Running simulation on 64 CPU cores can take typically $O(\text{days})$
  - How to speed this up? Days $\rightarrow$ minutes?
Hardware accelerators

GPUs are well equipped to train AI models

- Thousands of cores
- Large memory bandwidth
- Matrix multiplication

Since 2016 Google has launched **TPUs** specifically to increase AI performance → **Also great for HPC**
1 Cloud TPU has 4 chips
2 cores/chip
8 cores

256 Cloud TPUs form a v3 Pod
2048 cores
Distribution and Parallelization

Initialization and getting results through tpu.replicate input/output

Partitioned grid as variables collocated on HBM

Remote Program

initial values/results

Loading checkpoint/periodic output

TPU Chip 0

TPU Chip 1

TPU Chip 2

TPU Chip 3

ICI

Replicated compute

TPU Workers
# Simulation performance comparison

**Single CPU core vs. single TPU core**

Intel Xeon E5-16504 v4 @ 3.6 GHz vs. Google Cloud TPU v3

<table>
<thead>
<tr>
<th>Cell Size (m)</th>
<th>Grid Points (M)</th>
<th>CPU steps/s</th>
<th>TPU steps/s</th>
<th>Speed Up</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>62</td>
<td>0.26</td>
<td>30.22</td>
<td>118</td>
</tr>
<tr>
<td>8</td>
<td>15</td>
<td>1.04</td>
<td>118</td>
<td>114</td>
</tr>
</tbody>
</table>

4m resolution for 1.7 million steps:
77 days for 1 CPU core vs.
16 hours for 1 TPU core → 512 cores → **9 minutes**
Arkansas Flood Simulation Performance for 1 to 512 cores

The graph shows the performance of the simulation in terms of steps per second (steps/s) as a function of grid points per core. The data is split into four categories: 1m, 2m, 4m, and 8m, each represented by a different marker type.

- **1m** is represented by blue plus signs (+)
- **2m** is represented by orange crosses (x)
- **4m** is represented by green circles (o)
- **8m** is represented by red squares (■)

The x-axis represents the grid points per core, ranging from $10^5$ to $10^8$. The y-axis represents the steps per second, ranging from $10^1$ to $10^3$.
Layout of TPU Cores

- 2D simulation → 32 cores (e.g.) can have various assignments per axis: 1x32, 2x16, 4x8, etc.
- In many HPC settings, a more square per-core grid will be most efficient (8x4 in this case since the grid aspect ratio is ~2)
- TPUs have very high bandwidth, so latency dominates
  - The most extreme layouts (e.g. 32x1 or 1x32) are most efficient in this 2D case
### Time to compute 1.7 million steps

<table>
<thead>
<tr>
<th>Resolution</th>
<th>8 cores</th>
<th>32 cores</th>
<th>128 cores</th>
<th>512 cores</th>
</tr>
</thead>
<tbody>
<tr>
<td>8m</td>
<td>43 mins</td>
<td>13 mins</td>
<td>5.9 mins</td>
<td>6.1 mins</td>
</tr>
<tr>
<td>4m</td>
<td>2.7 hours</td>
<td>44 min</td>
<td>15 mins</td>
<td>8.9 mins</td>
</tr>
<tr>
<td>2m</td>
<td>10 hours</td>
<td>2.7 hours</td>
<td>46 mins</td>
<td>18 mins</td>
</tr>
<tr>
<td>1m</td>
<td>40 hours</td>
<td>10 hours</td>
<td>2.7 hours</td>
<td>53 mins</td>
</tr>
</tbody>
</table>

1.728 million steps = 2 simulation days if $\Delta t = 0.1$ sec.
<table>
<thead>
<tr>
<th>Resolution</th>
<th>8 cores</th>
<th>32 cores</th>
<th>128 cores</th>
<th>512 cores</th>
</tr>
</thead>
<tbody>
<tr>
<td>8m</td>
<td>43 mins</td>
<td>13 mins</td>
<td>5.9 mins</td>
<td>6.1 mins</td>
</tr>
<tr>
<td>4m</td>
<td>2.7 hours</td>
<td>44 mins 97%</td>
<td>15 mins 84%</td>
<td>8.9 mins 66%</td>
</tr>
<tr>
<td>2m</td>
<td>10 hours</td>
<td>2.7 hours 99%</td>
<td>46 mins 93%</td>
<td>18 mins 72%</td>
</tr>
<tr>
<td>1m</td>
<td>40 hours</td>
<td>10 hours 100%</td>
<td>2.7 hours 100%</td>
<td>53 mins 80%</td>
</tr>
<tr>
<td>Resolution</td>
<td>8 cores</td>
<td>32 cores</td>
<td>128 cores</td>
<td>512 cores</td>
</tr>
<tr>
<td>------------</td>
<td>---------</td>
<td>----------</td>
<td>-----------</td>
<td>-----------</td>
</tr>
<tr>
<td>8m</td>
<td>43 mins</td>
<td>13 mins 83%</td>
<td>5.9 mins 46%</td>
<td>6.1 mins 11%</td>
</tr>
<tr>
<td>4m</td>
<td>2.7 hours</td>
<td>44 mins 91%</td>
<td>15 mins 66%</td>
<td>8.9 mins 28%</td>
</tr>
<tr>
<td>2m</td>
<td>10 hours</td>
<td>2.7 hours 94%</td>
<td>46 mins 83%</td>
<td>18 mins 54%</td>
</tr>
<tr>
<td>1m</td>
<td>40 hours</td>
<td>10 hours 99%</td>
<td>2.7 hours 94%</td>
<td>53 mins 70%</td>
</tr>
</tbody>
</table>
Flood forecasting using hydraulic models

● In **steady-state** rivers, many simulations with different discharges are typically done offline, before flood season.

● During flooding, given actual and predicted stream gauge measurements, the correct discharge is picked out and alerts are sent out.

● Changing run times from days to minutes allows for a real-time approach.

● Also, real time approaches are needed in case of dynamic rivers (**non-steady-state**).
Conclusion

● Hydraulic flood simulations are a useful tool in flood forecasting

● Running simulations on TPUs can dramatically decrease run times
  ○ Scaling results shown for Arkansas flood simulation

● Running on a fleet of TPUs opens the possibility for real time approaches in both steady-state and dynamic river cases (e.g. variational data assimilation)

● AI: TPUs can readily generate data sets for machine learning training

● Paper in progress; GCP Python interactive notebook with flood simulation will be made available