Using MapD's GPU-powered SQL Database to Interact with National Water Model (NWM) Predictions

Aaron Williams, *MapD Technologies*
Devika Kakkar, *Harvard Center for Geographic Analysis (CGA)*

2018 CGA Conference: Illuminating Space and Time in Data Science
April 26, 2018 to April 27, 2018

Center for Geographic Analysis
Harvard University

Spatiotemporal Thinking, Computing and Applications
STC

NSF

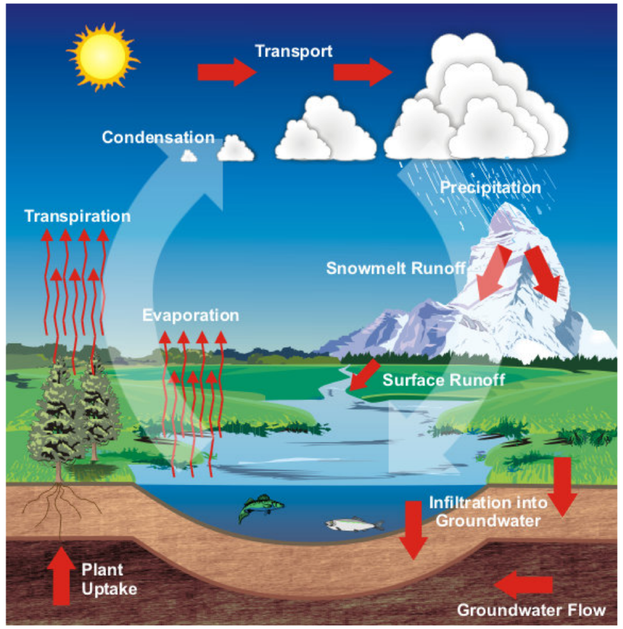
Abstract

- In this workshop, we will show how CGA researchers are using MapD’s open-source, GPU-powered SQL database to provide true interactive access to NWM predictions for stream flow and ground saturation across the entire continental US, from present conditions to 18 days in the future.
- Predictions can be viewed prospectively, “how will conditions change going forward?” as well as retrospectively, “how did condition predictions evolve up to any given present?”.
- Water conditions can also be tracked in space and time together as storms move across the country.

Workshop Outline

- The National Water Model
- Data Flow from NWM to MapD
- Memory management MapD
- Present work with NWM and MapD
- Time Perspective with NWM and MapD
- Visualization in MapD
  - Table optimization
  - Database switches and memory
- Conclusions

The National Water Model



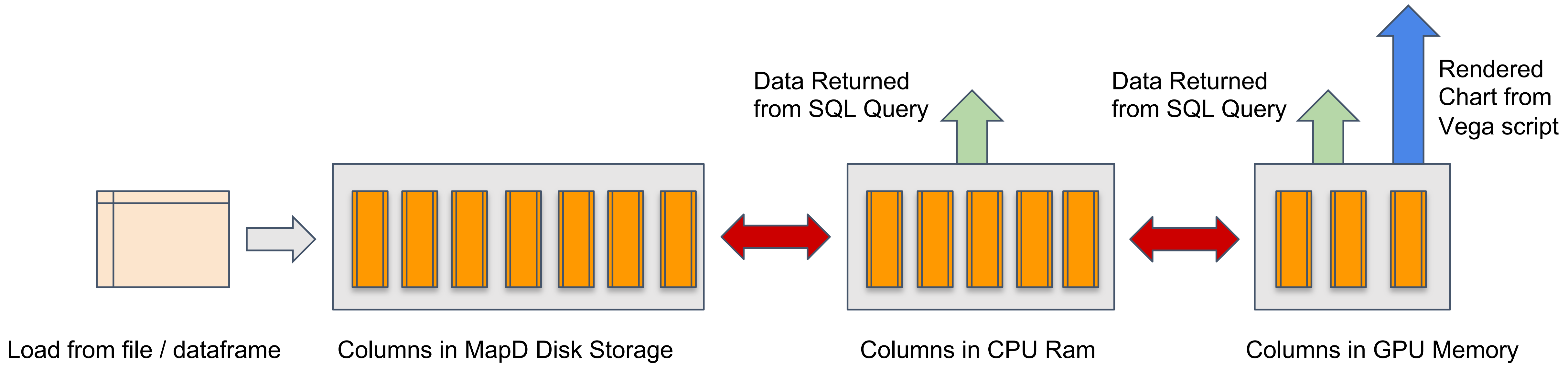
- U.S. National Water Model (NWM) models run up to hourly on a Cray XC40 supercomputer.
- Input data from ~3600 river / reservoir gauges, along with weather model outputs and other data sources, generates predictions (present, 0-18-hr, 0-10-day, or 0-30-day) of hydrologic conditions
- Predictions for 2.7 million stream reaches, 1260 reservoirs, and at ~300M surface grid points across the U.S. (1km & 250m spacings).
- NWM outputs ~90gb / day (1gb present conditions, 18gb shortrange, 65gb / day medium range, ~4gb / day long range).
- A viewer is available for pre-generated images of present model output and another for pre-generated grouped streamflow features.

Data Flow from NWM to MapD

- Harvesting
  - NWM output files in NetCDF format downloaded from website.
- Storage
  - 2-month test dataset at 6-hour intervals of present and predicted conditions stored in AWS S3.
  - Next: a rolling 1-2 month time window of datafiles to be maintained on the MOC
- Preprocessing / loading to MapD
  - NetCDF -> Xarray -> Pandas Dataframe -> PyMapD -> MapD table
  - Geometric coordinates stored separately from model output parameters
- Limitations
  - Data are initially loaded onto disk, then column-wise into (limited) CPU and/or GPU memory for query and/or rendering (1 K-80 GPU -> ~11gb data memory).

How MapD manages memory spaces

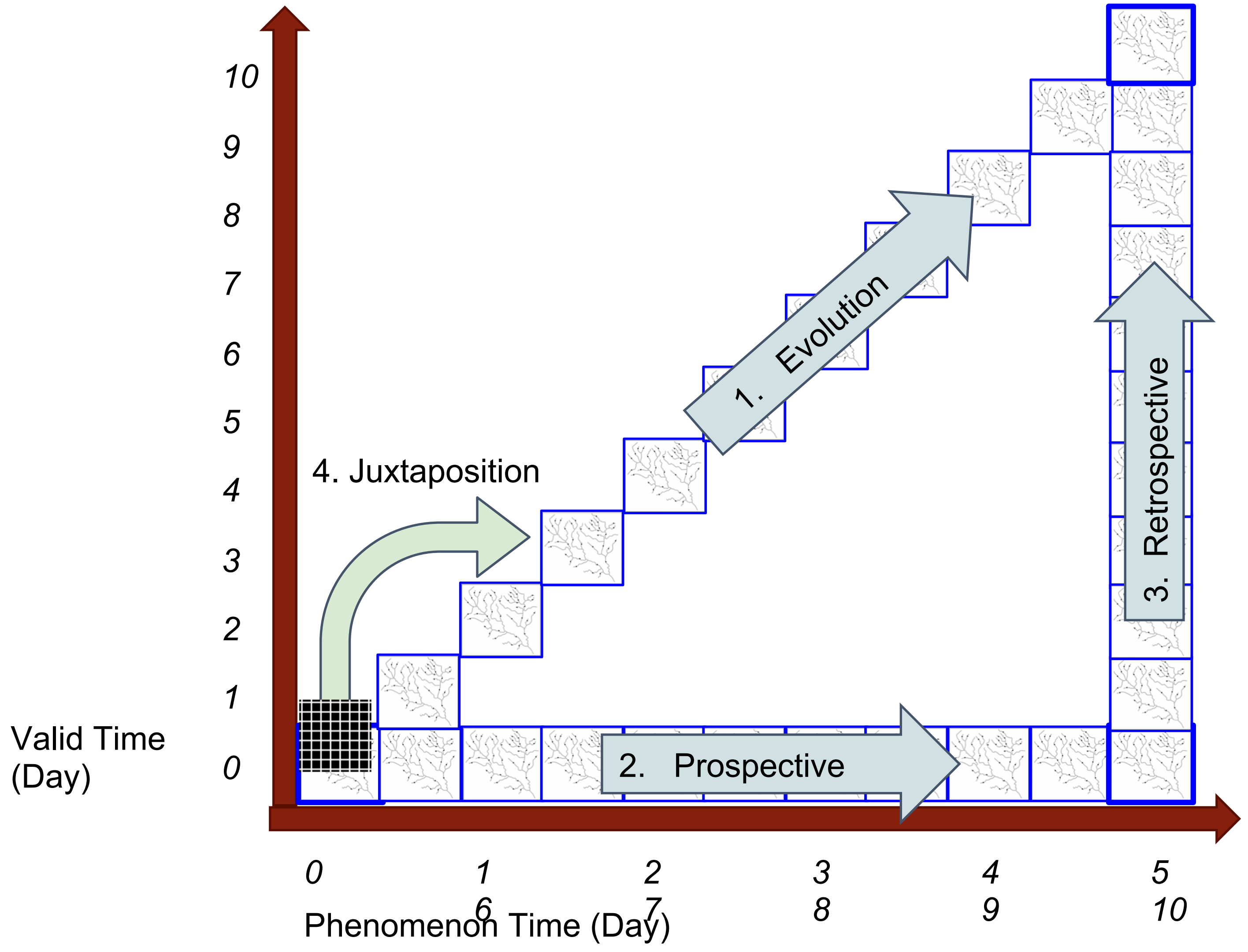
- Datasets are initially loaded column by column into disk files
- Columns are loaded into CPU memory (RAM) and then GPU memory as available to execute queries that address those columns. Additional memory space is used for any intermediate data structures and/or query results.
- GPU memory is also used to render chart and map images from query results


Data Returned from SQL Query
Data Returned from SQL Query
Rendered Chart from Vega script
Load from file / dataframe
Columns in MapD Disk Storage
Columns in CPU Ram
Columns in GPU Memory

Present work with NWM and MapD

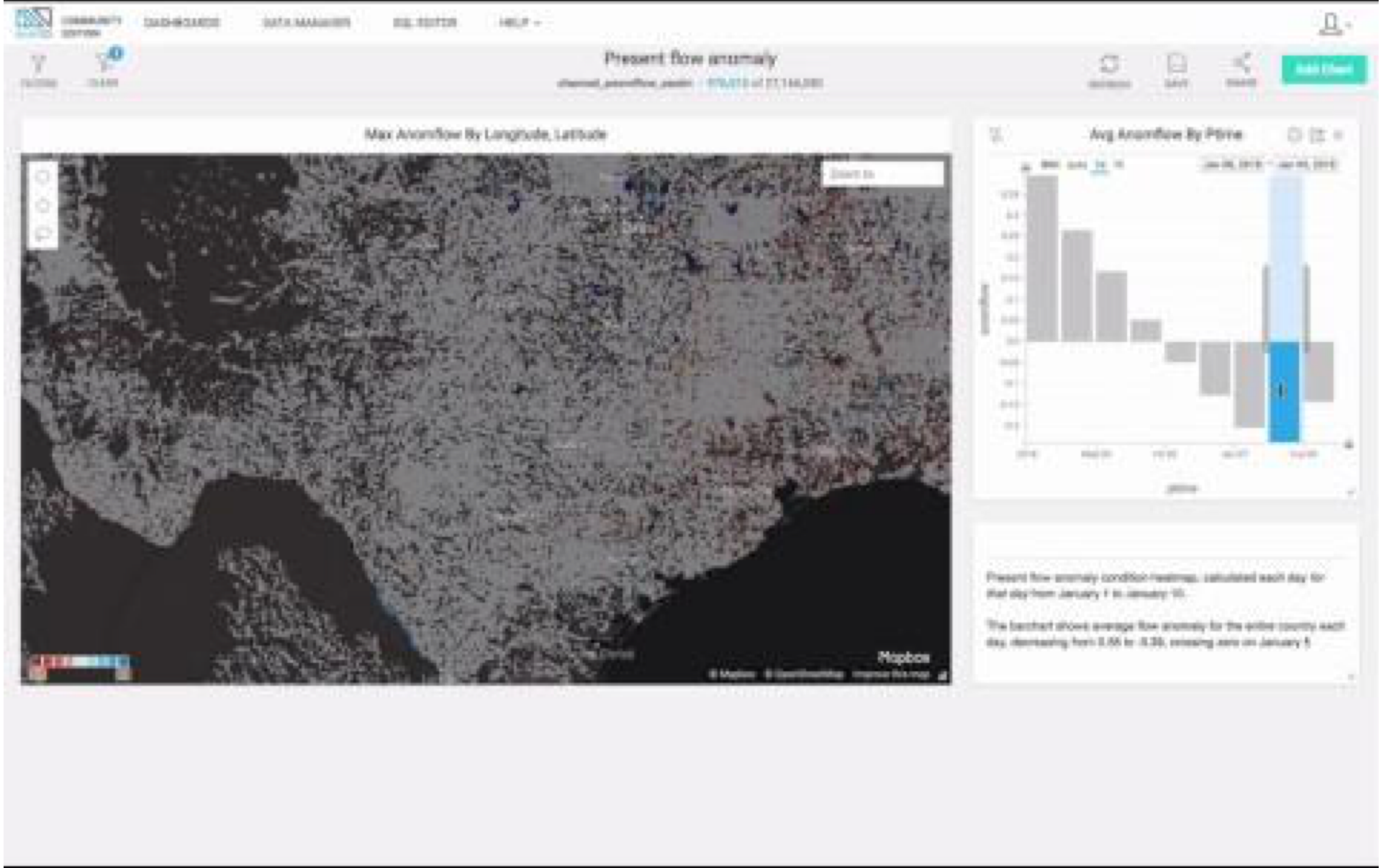
- Develop data download, storage, database loading procedures
- Configure and install MapD on 1-GPU EC2 instances
- Load nationwide stream reach mid-points and 1km (down-sampled from 250m to fit in GPU memory) grid point geometries to MapD store.
- Load stream flow / velocity and soil inundation / saturation outputs for once-daily present conditions and 1-10 day predictions over a 10-day period at the beginning of 2018. Learn virtues of pandas dataframe.
- Construct SQL views to join stream and point locations with model output values. Discover that (some) views work differently than equivalent queries in MapD.
- Develop dashboard views in MapD Immerse to visualize the results. Learn hidden tricks, GPU utilities, and undocumented configuration switches.

Time perspectives in NWM output data



1. Evolution of present conditions over 10 days
2. Prospective prediction over 10 days from start of period
3. Retrospective evolution of 10th day predictions from 0-day prediction to present conditions
4. Juxtaposition and time offset for influence of surface / subsurface flow routing on nearby river inputs and flow (not shown in this presentation)

Live queries bounded by map area



Present flow anomaly condition heatmap, calculated each day for that day from January 1 to January 10.

The barchart shows average flow anomaly for the entire country each day, decreasing from 0.55 to -0.39, crossing zero on January 5.

Table optimization

- **Issues**

Database switches and memory

- Issues with views that have joins

create table channel_flow_anomaly as select a.feature_id, b.longitude, b.latitude, (a.streamflow - b.avgflow) as anomflow, a.valid_time as vtime, a.phenomenon_time as ptime from nwm_channel a join channel_coord_avg b on a.feature_id=b.feature_id;

Exception: Query couldn't keep the entire working set of columns in GPU memory

- The following switches were added to optimize the memory usage:
  - enable-watchdog = false: Allows the query to run in stages on the GPU MapD will then orchestrate the transfer of data through the various layers of our data abstraction to move the data onto the GPU for execution
  - allow-cpu-retry = true: Allows to direct queries that do not fit in the GPU memory available, to fall back and be executed on the CPU

Conclusions

- Interpretation of simulation / prediction model outputs for geographic entities can be a big data challenge that is both significant and separate from that of running the models. Without adequate tools to interpret this scale of data, the usefulness of creating and running the models themselves is reduced.
- GPU-based data analysis and visualization tools such as MapD offer good possibilities for addressing this challenge with fast data interaction, cost effective deployment, and flexible integration with other tools.
- DBMS' such as MapD still require significant expertise to use effectively when "pushing the envelope" on new capabilities.
- CGA has learned much already from working with MapD and NWM model outputs and plan to apply this to other use cases and domains.

Links

- More detail on the Project Wiki https://github.com/cga-harvard/HPC_on_MOC/wiki
- MapD Core code https://github.com/mapd/mapd-core
- Collaboration announcements http://gis.harvard.edu/announcements/renewed-collaboration-between-cga-and-mapd-accelerate-research-gpus

Thank You