DEVELOPMENT OF A KNOWLEDGE-SHARING PARALLEL COMPUTING APPROACH FOR CALIBRATING DISTRIBUTED WATERSHED HYDROLOGIC MODELS

MARJAN ASGARI

UNIVERSITY OF GUELPH

Computationally expensive Calibration of Distributed Hydrological Models

Watershed hydrologic models developed to study landscape processes such as runoff, sediment, and nutrient processes.

Computationally-expensive calibration process:

1- Extensive spatiotemporal heterogeneity of predominant hydrological processes.

2- Applying global (meta)heuristic optimization algorithms, which imposes high computational requirement.

3- Using high-resolution datasets and/or working on larger watersheds.

Parallel computing has been increasingly applied to address the computational challenges of calibrating watershed hydrologic models.

The Existing Literature in the Parallel Computing Application in Calibration of Hydrologic Models

The Lack of

1- Knowledge-Sharing Among Parallel Computing Units

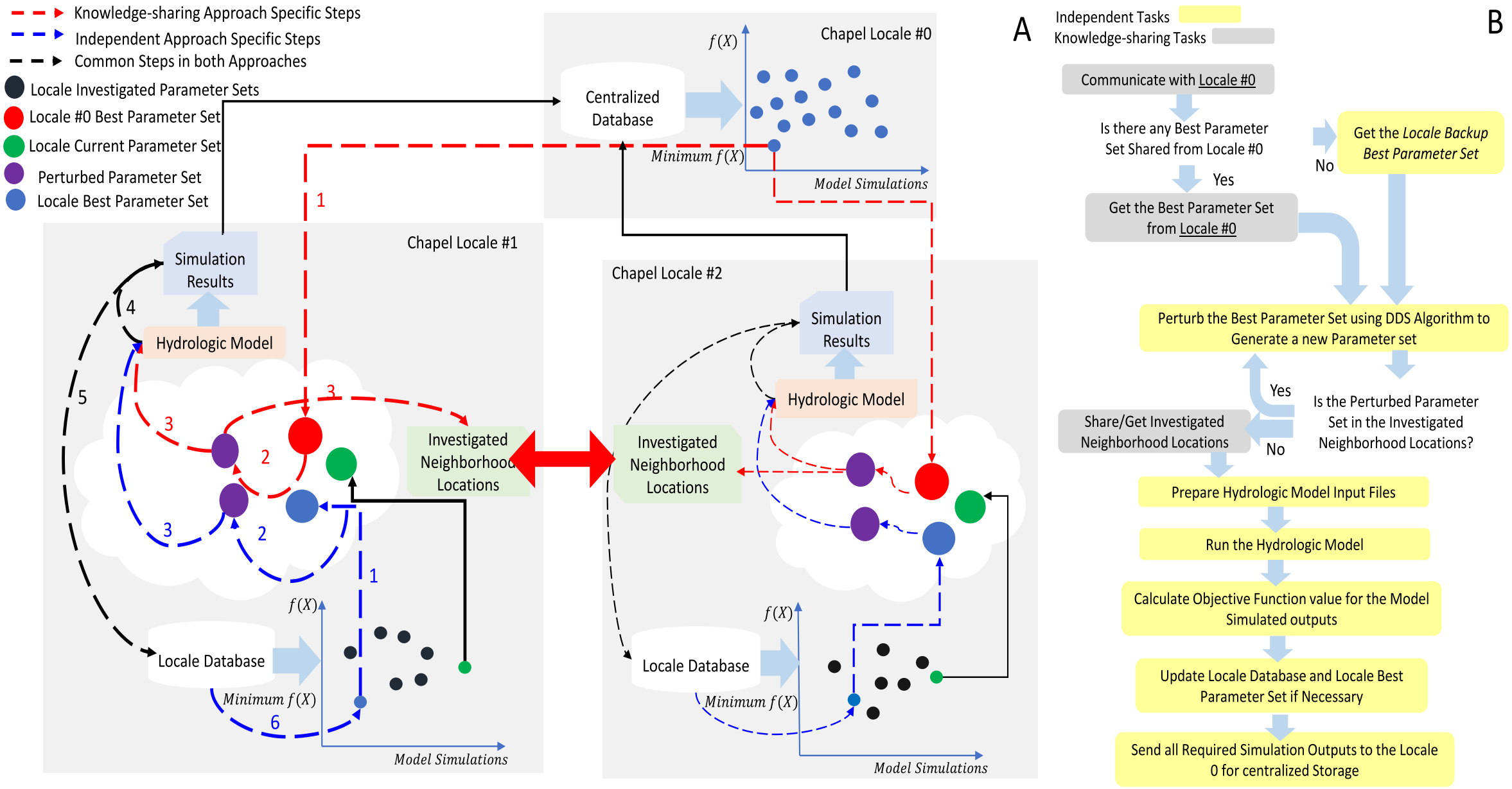
2- High programmability of used frameworks/languages.

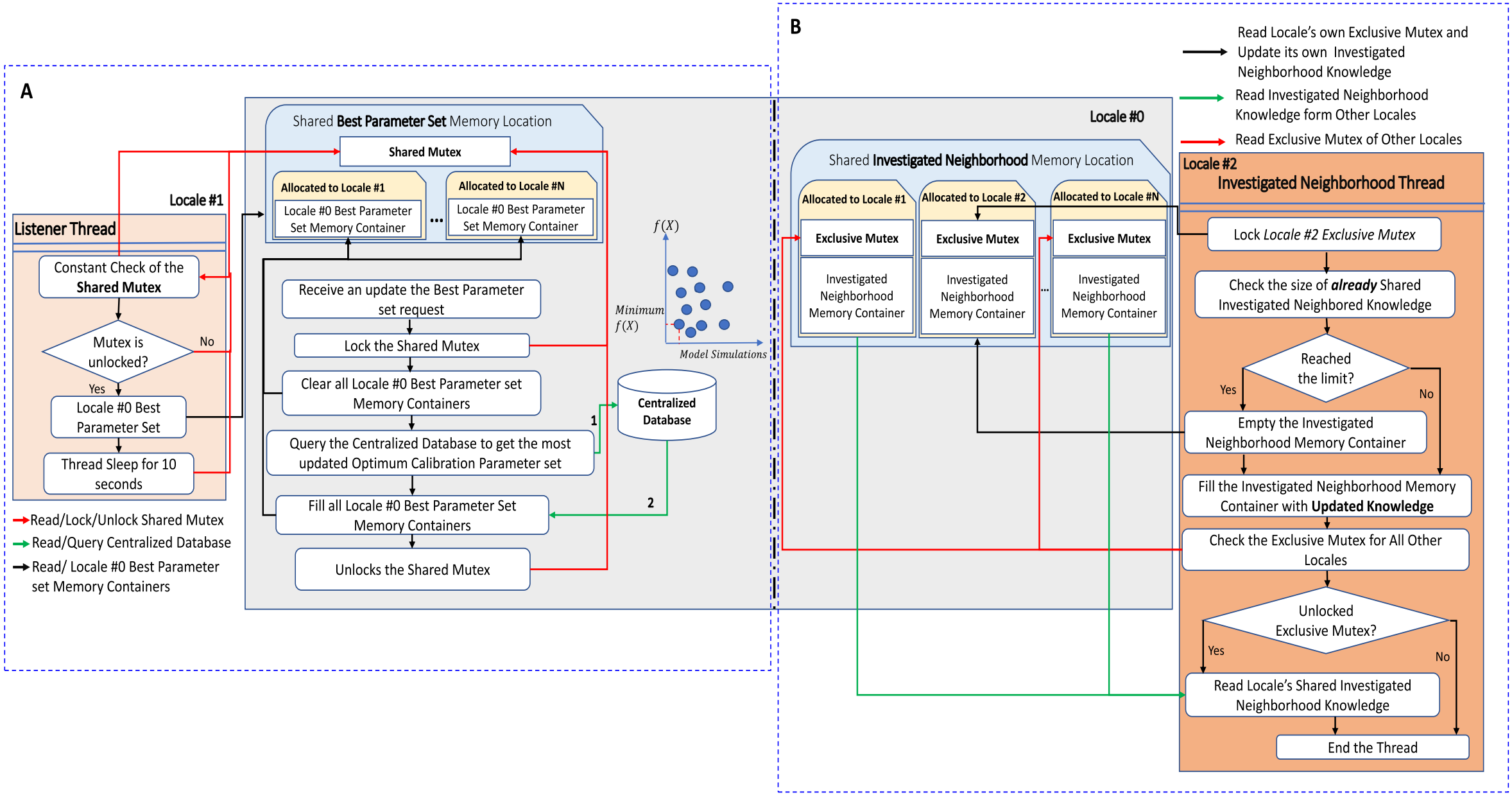
3- High Parallel Efficiency

4- Capability of handling failed nodes

…

Partitioned Global Address Space (PGAS) paradigm had not been implemented .
Thus we used Chapel Programming Language which implements PGAS to address the current research gaps.


Knowledge-sharing Approach Specific Steps
Independent Approach Specific Steps
Common Steps in both Approaches
Locale Investigated Parameter Sets
Locale #0 Best Parameter Set
Locale Current Parameter Set
Perturbed Parameter Set
Locale Best Parameter Set
Chapel Locale #0
Centralized Database
f(X)
Minimum f(X)
Model Simulations
Chapel Locale #1
Simulation Results
Hydrologic Model
Investigated Neighborhood Locations
Locale Database
Chapel Locale #2
A
Independent Tasks
Knowledge-sharing Tasks
B
Communicate with Locale #0
Is there any Best Parameter Set Shared from Locale #0
Yes
No
Get the Locale Backup Best Parameter Set
Get the Best Parameter Set from Locale #0
Perturb the Best Parameter Set using DDS Algorithm to Generate a new Parameter set
Is the Perturbed Parameter Set in the Investigated Neighborhood Locations?
Share/Get Investigated Neighborhood Locations
Prepare Hydrologic Model Input Files
Run the Hydrologic Model
Calculate Objective Function value for the Model Simulated outputs
Update Locale Database and Locale Best Parameter Set if Necessary
Send all Required Simulation Outputs to the Locale 0 for centralized Storage


A
Shared Best Parameter Set Memory Location
Shared Mutex
Allocated to Locale #1
Locale #0 Best Parameter Set Memory Container
Allocated to Locale #N
Locale #0 Best Parameter Set Memory Container
Locale #1
Listener Thread
Constant Check of the Shared Mutex
Mutex is unlocked?
No
Yes
Locale #0 Best Parameter Set
Thread Sleep for 10 seconds
Receive an update the Best Parameter set request
Lock the Shared Mutex
Clear all Locale #0 Best Parameter set Memory Containers
Query the Centralized Database to get the most updated Optimum Calibration Parameter set
Fill all Locale #0 Best Parameter Set Memory Containers
Unlocks the Shared Mutex
Centralized Database
1
2
f(X)
Minimum f(X)
Model Simulations
Read/Lock/Unlock Shared Mutex
Read/Query Centralized Database
Read/ Locale #0 Best Parameter set Memory Containers
B
Read Locale's own Exclusive Mutex and Update its own Investigated Neighborhood Knowledge
Read Investigated Neighborhood Knowledge form Other Locales
Read Exclusive Mutex of Other Locales
Locale #0
Shared Investigated Neighborhood Memory Location
Allocated to Locale #1
Exclusive Mutex
Investigated Neighborhood Memory Container
Allocated to Locale #2
Exclusive Mutex
Investigated Neighborhood Memory Container
Allocated to Locale #N
Exclusive Mutex
Investigated Neighborhood Memory Container
Locale #2
Investigated Neighborhood Thread
Lock Locale #2 Exclusive Mutex
Check the size of already Shared Investigated Neighbored Knowledge
Reached the limit?
Yes
No
Empty the Investigated Neighborhood Memory Container
Fill the Investigated Neighborhood Memory Container with Updated Knowledge
Check the Exclusive Mutex for All Other Locales
Unlocked Exclusive Mutex?
Yes
No
Read Locale's Shared Investigated Neighborhood Knowledge
End the Thread

The Challenges of Chapel Q-threads Tasking Layer for Running Hydrologic Models

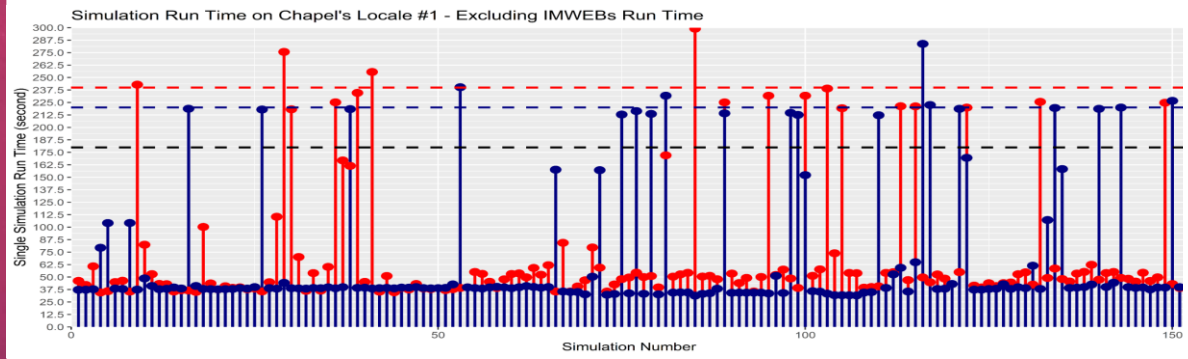
In studies with the CPU-intensive part of the program being carried out in Chapel sub-processes, the *QThreads CPU pinning* behavior should be modified to leave some idle CPUs just for running the external models.

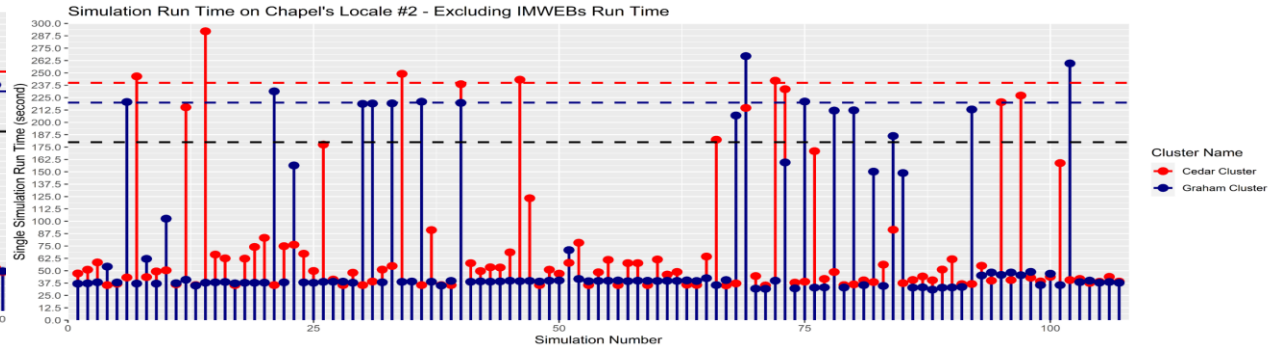
> Disabling QThreads CPU pinning, which in theory, allows the Chapel threads and the subprocess threads to migrate away from one another.

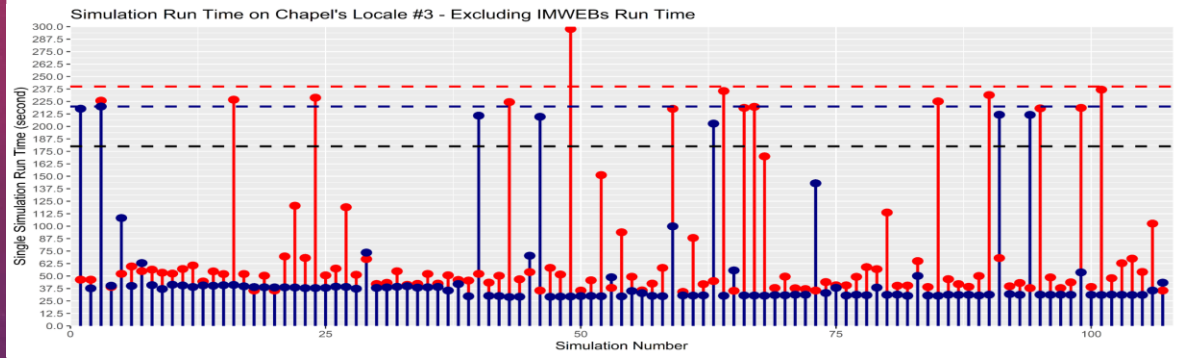
Setting ***QT_AFFINITY=no*** and ***CHPL_RT_NUM_THREADS_PER_LOCALE***

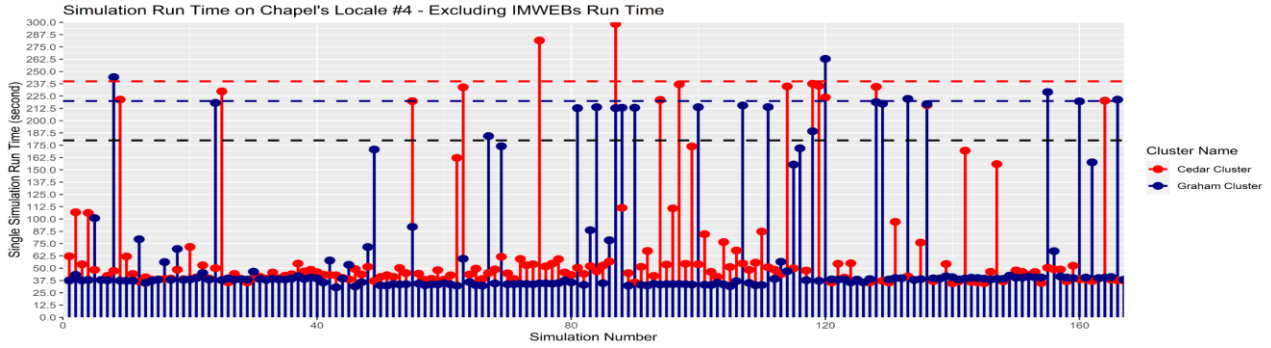
In our research, setting these two variables led to superb performance improvement in running our hydrologic model; the runtime of a single IMWEBs model simulation for an 8-year calibration period decreased from 65 minutes to 13 minutes on a Locale with 12 CPU cores.

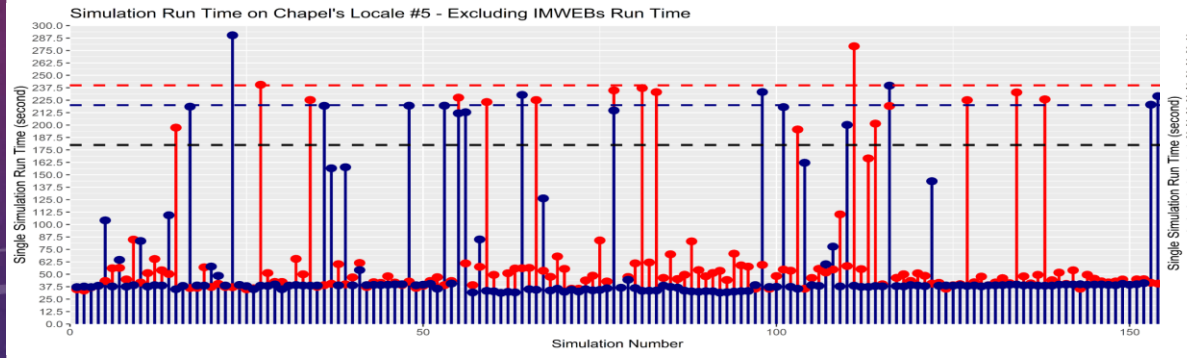
Some of our Results

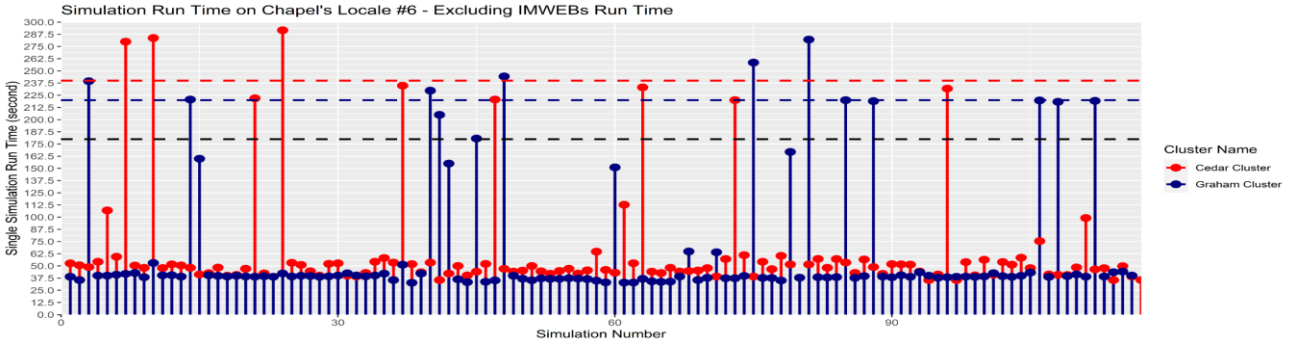




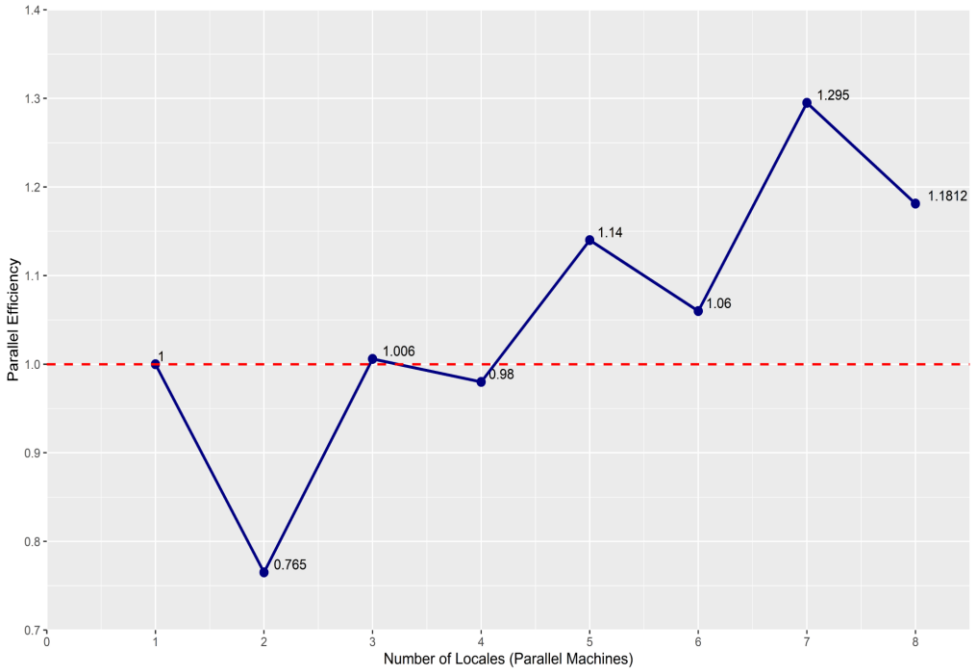


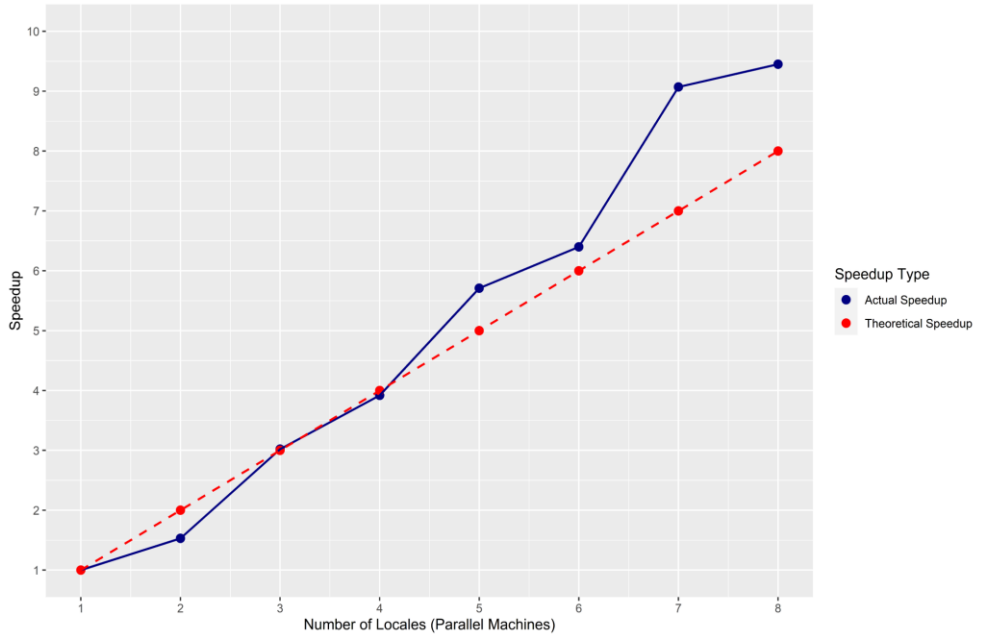






Some of our Results





THANK YOU!

MASGARI@UOGUELPH.CA