APPFIS: An Advanced Parallel Programming Framework for Iterative Stencil Based Scientific Applications in HPC Environments

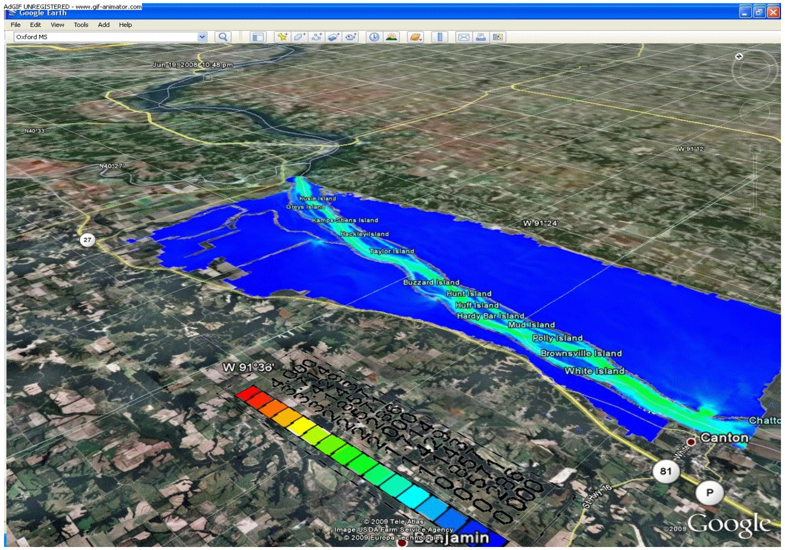
Sheikh Ghafoor

July 11, 2022

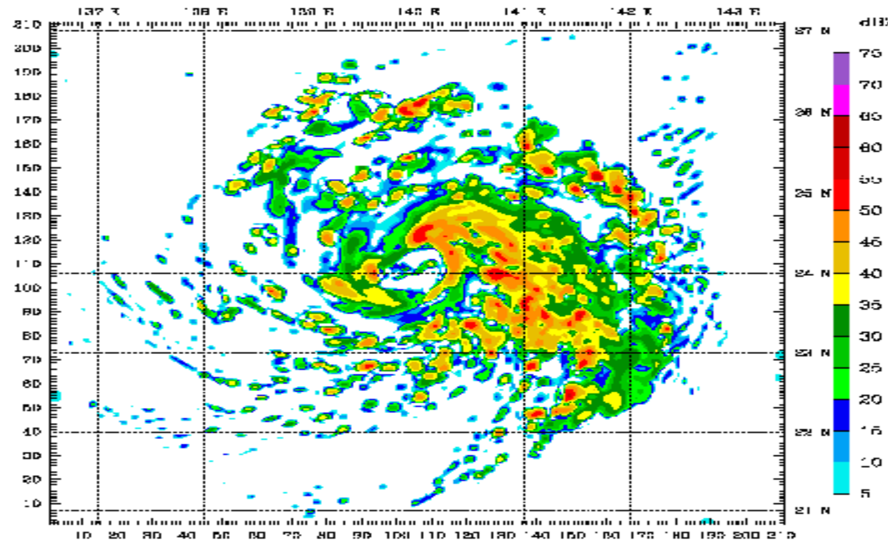
Overview

- Motivation
- Related Work
- Methodology
- Result and Analysis
- Conclusion

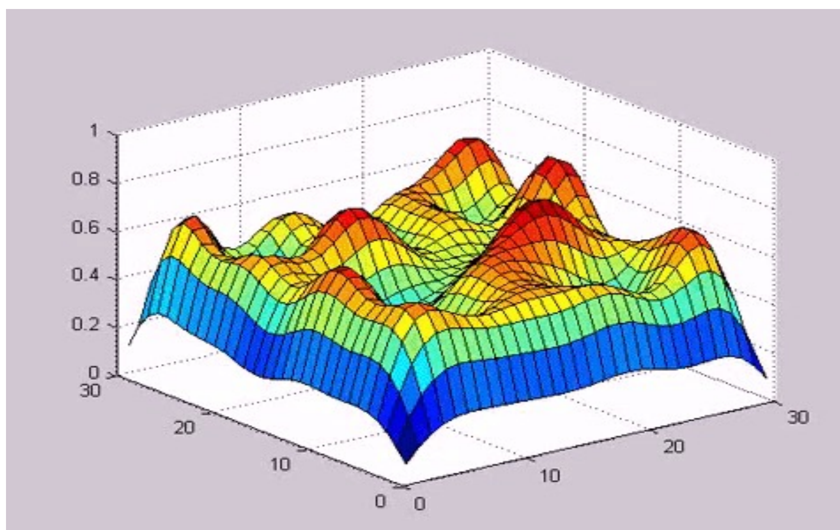
Introduction



Flood

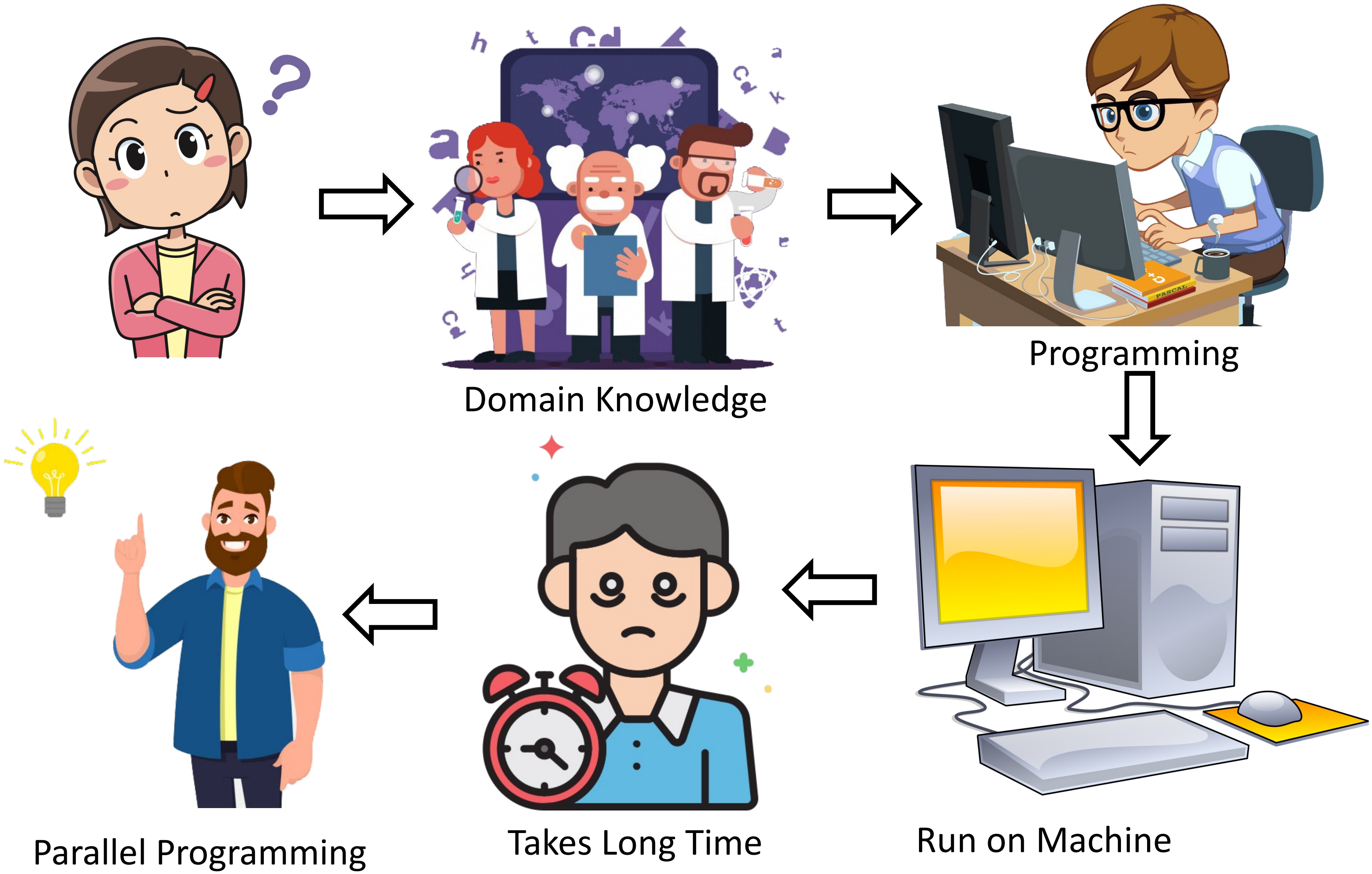


Weather

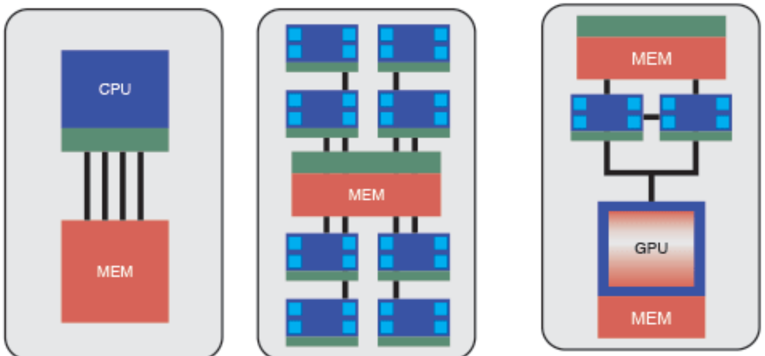


Heat

Scientific Application Development



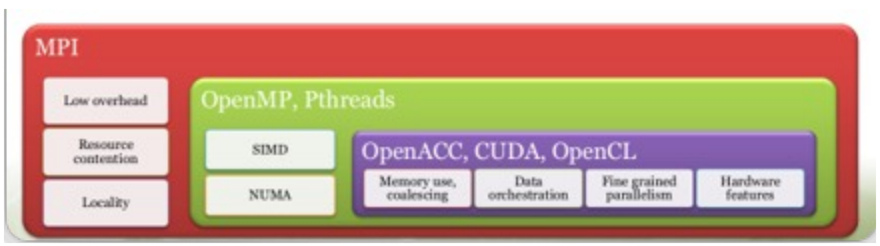
High Performance Computing

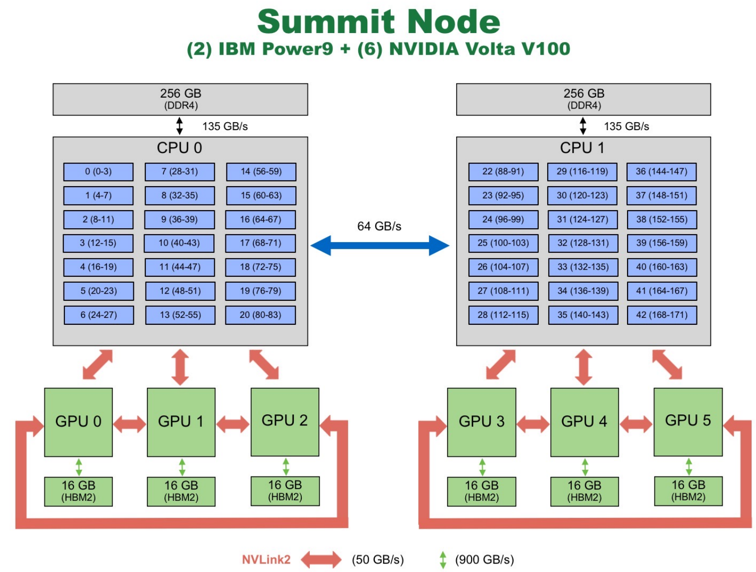


1995
Single CPU per node
with main memory

2000–2010
Multiple CPUs per node
sharing main memory

2000–2010
Accelerators usher in
era of heterogeneity



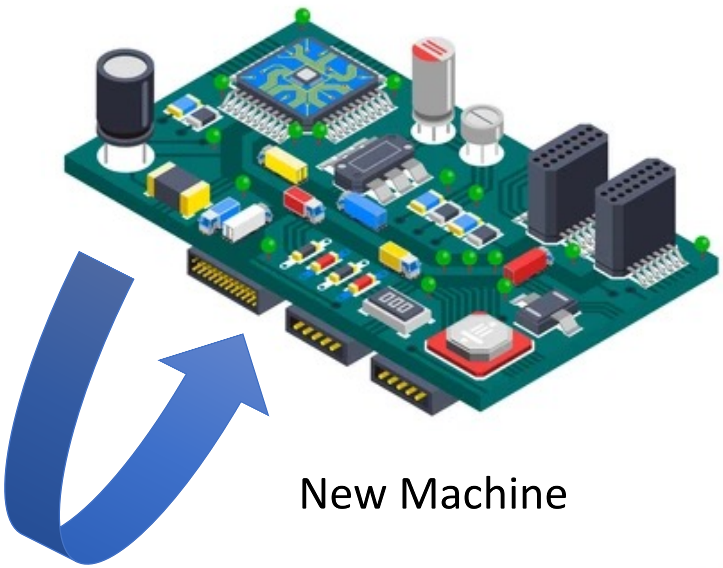


Time

Money

Expertise

Problem



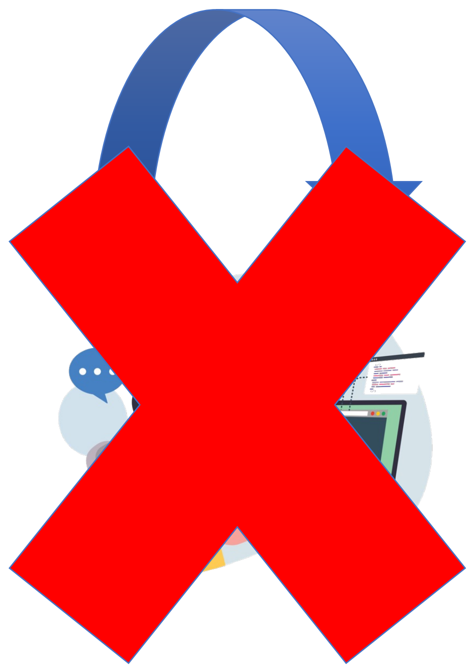
New Machine

Summary

Domain Expert

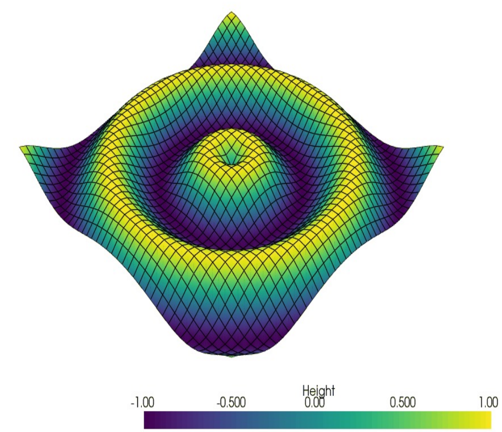
Programming Knowledge

Repeat...

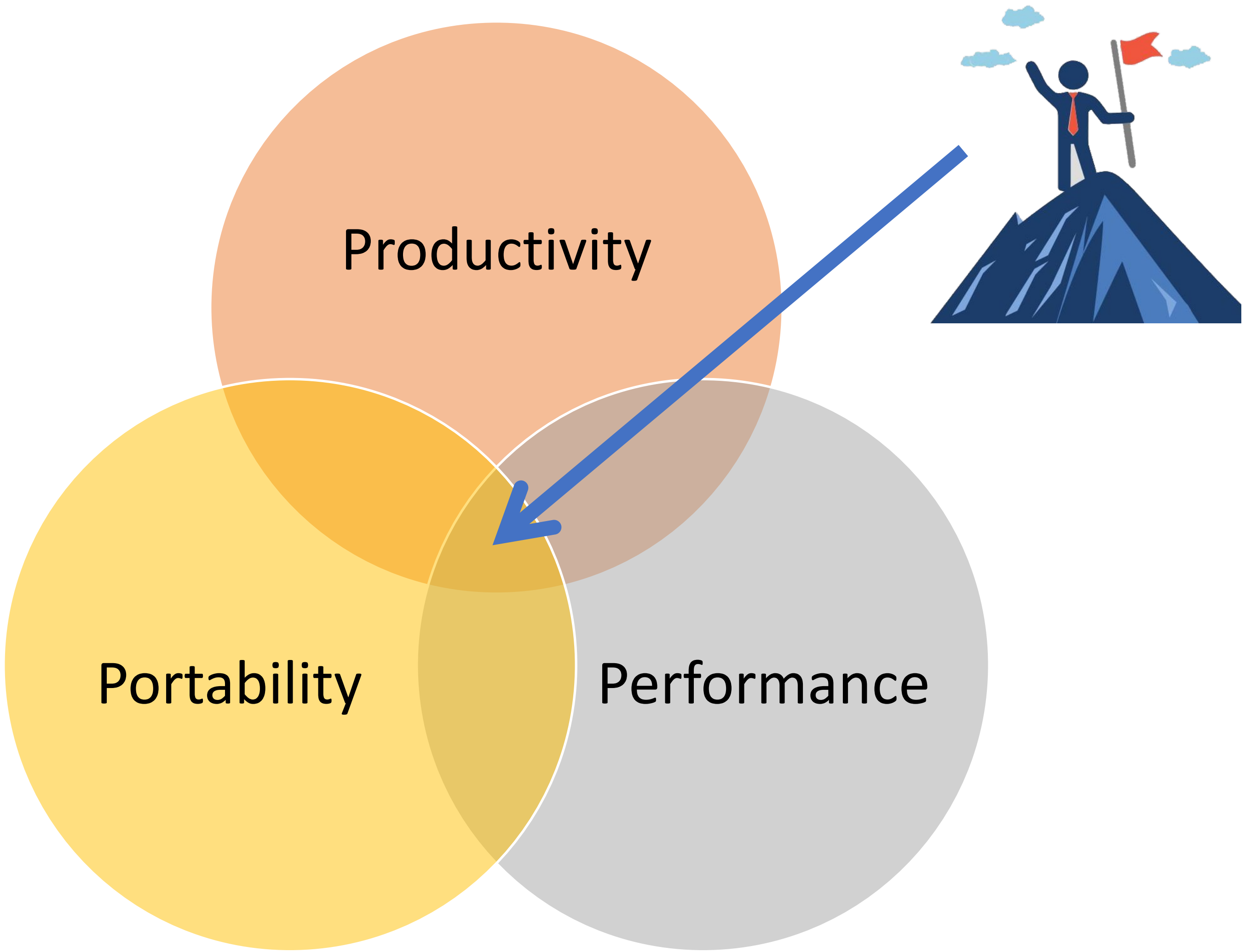


HPC Expert

Efficient Application



P, P and P



Related Works

- Physis [1], PARTANS [2], PATUS [3], Panda [4], STELLA [5] and Pskel [6] are some existing framework also focused on stencil applications.
- Kokkos [7] and RAJA [8] are the most popular performance portable programming model for manycore devices.
- Most of the existing frameworks lack multi GPU support and hybrid heterogeneous computation.
- Lacking of multi dimension partition, different boundary condition and reduction operation.
- Some of the frameworks use overlapping between computation and communication. But only for CPU + GPU case.
- No measurement and comparison between frameworks in terms performance portability.
- No formal definition, measurement and analysis of productivity.
- Lack of real application development and performance measure.
- Not open sourced

Comparison (1/3)

Name	Physis	Partans	Patus	Panda	Stella	Pskel	Kokkos
Code Translation	yes	no	yes	yes	yes	no	no
Multi GPU	yes	yes	no	yes	no	no	no
OpenMP	no	no	yes	yes	yes	yes	yes
Overlapping	yes	yes	no	yes	no	no	no
Multi Dimention Partition	yes	no	no	yes	no	no	no
Multi Dimention	yes	yes	yes	no	no	yes	yes
Open Source	yes	no	yes	no	no	yes	yes
Library Independent	no	no	yes	no	yes	no	no
CPU-GPU Parallel	no	no	no	yes	no	yes	no

Comparison (2/3)

Name	Physis	Partans	Patus	Panda	Stella	Pskel	Kokkos
Boundary Condition	yes	no	yes	yes	yes	no	no
Struct Support	no	yes	no	no	no	no	no
Interior Cell Check Optimize	yes	no	no	no	no	no	no
Multiple Application	yes	yes	yes	yes	no	yes	no
Lines of code	yes	no	no	no	no	no	no
Variable Block Size	no	no	no	no	no	yes	no
Multiple Computation	no	yes	yes	no	no	no	no
Auto Partition	no	yes	no	yes	no	no	no
Optimal Halo Size	no	yes	no	no	no	no	no

Comparison (3/3)

Name	Physis	Partans	Patus	Panda	Stella	Pskel	Kokkos
GPU Selection	no	yes	no	no	no	no	no
Data transformation	no	yes	no	no	no	no	no
GPU Direct	no	no	no	no	no	no	no
Combine Boundary as Pack	no	no	no	no	no	no	no
Cache optimization	no	no	yes	no	yes	no	no
Reduction	no	no	no	yes	no	no	yes
Hybrid	no	no	no	yes	no	no	no
Performance Portability	no	no	no	no	no	no	yes
Productivity	yes	no	no	no	no	no	no

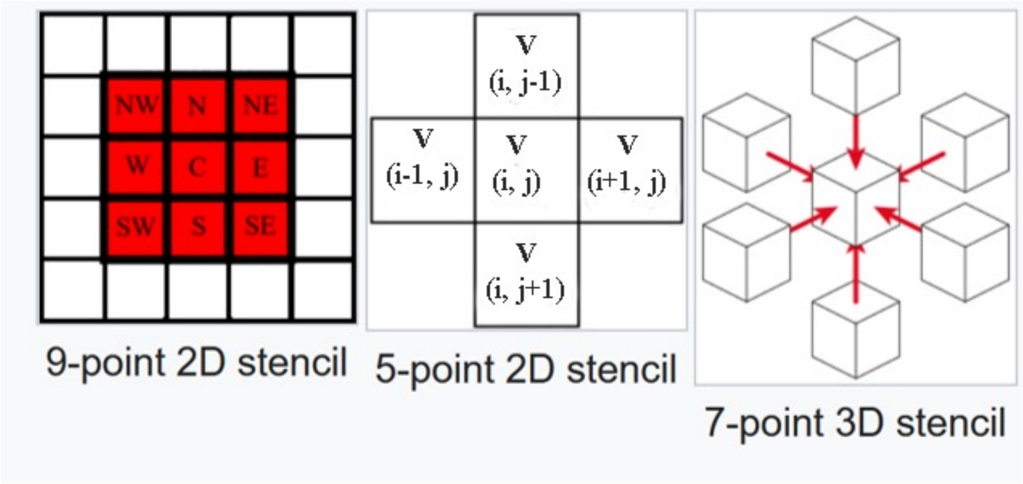
Stencil Computation

- Multi dimensional grid on neighboring values using a fixed pattern (Stencil).
- Example, heat transfer, flood simulation, weather forecasting, & many more

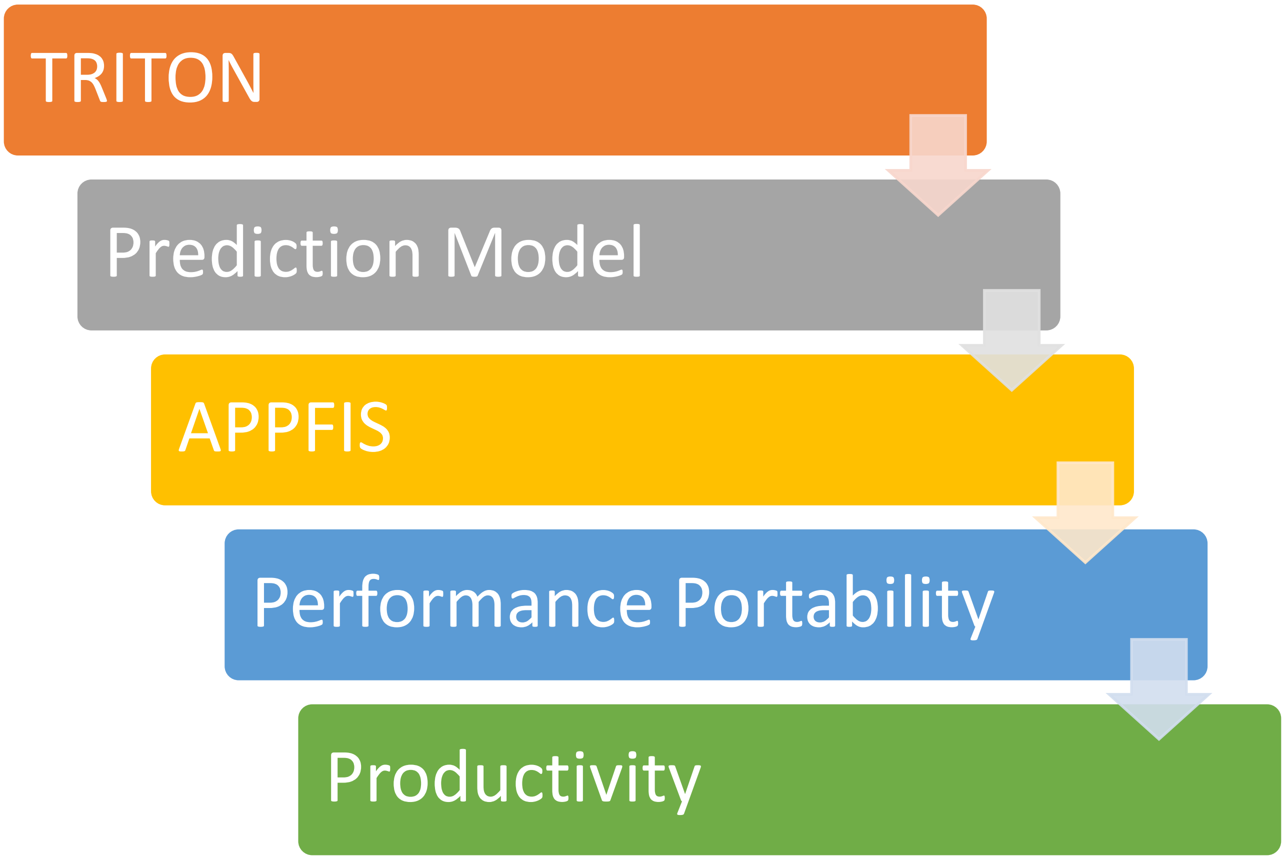
$$V^{n+1}(i,j) = f(V^n(i+\alpha, j+\beta)|\alpha, \beta \in C)$$ [1]

- 5 point diffusion stencil

$$V^{n+1}(i,j) = 1/5(V^n(i,j) + V^n(i+1,j) + V^n(i-1,j) + V^n(i,j+1) + V^n(i,j-1))$$ [1]

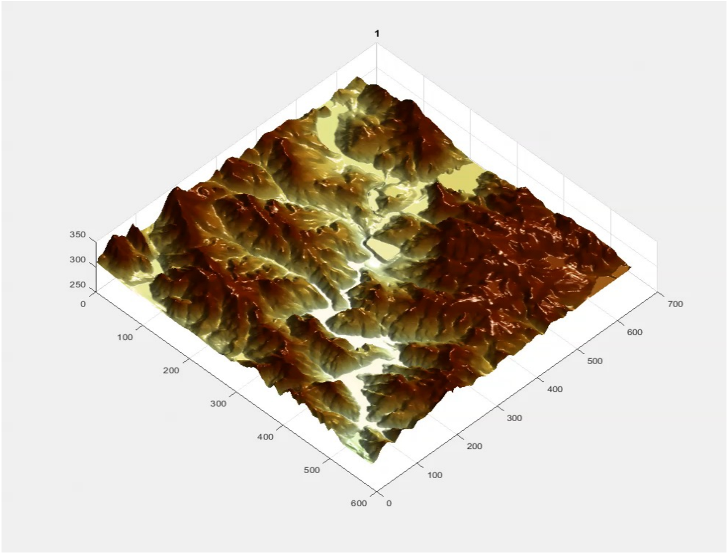


Methodology



TRITON (1/4)

- Two-dimensional hydrodynamic flood model with two different numerical schemes.
- There are two objectives for this effort
  - Develop an optimized implementation of a real application on multiple platforms to serve as a baseline
  - Gather experience on what is involved in developing structured grid-based stencil applications
- Four different test cases were simulated on the Summit supercomputer
- Numerical model proposed by Kalyanapu et al. [13]



TRITON (2/4)

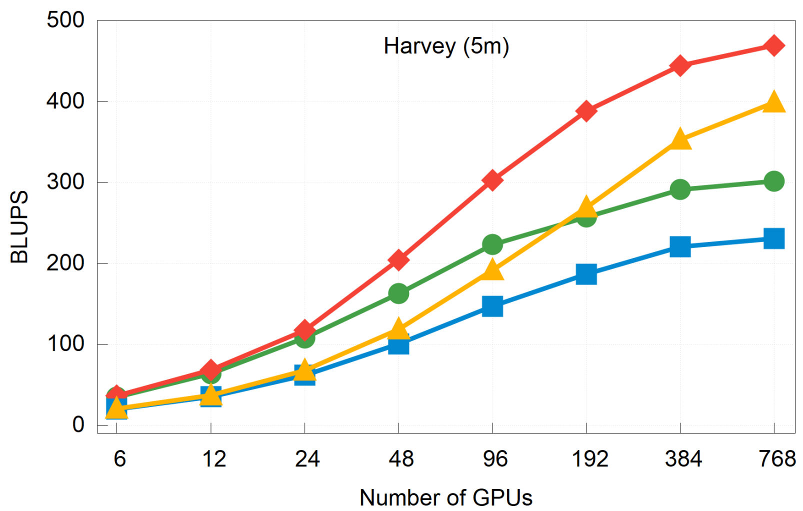


Fig: Strong Scaling

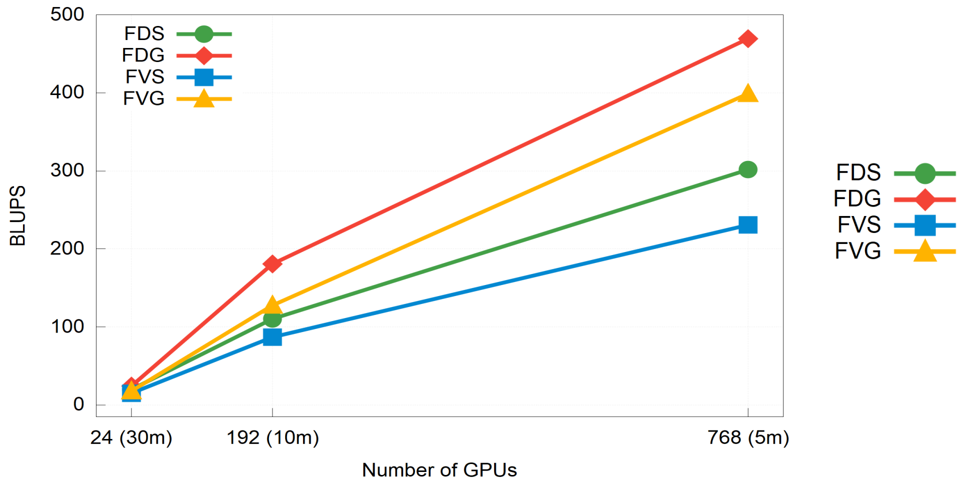


Fig: Weak Scaling

D: Difference
V: Volume
G: GPUDirect
S: Standard

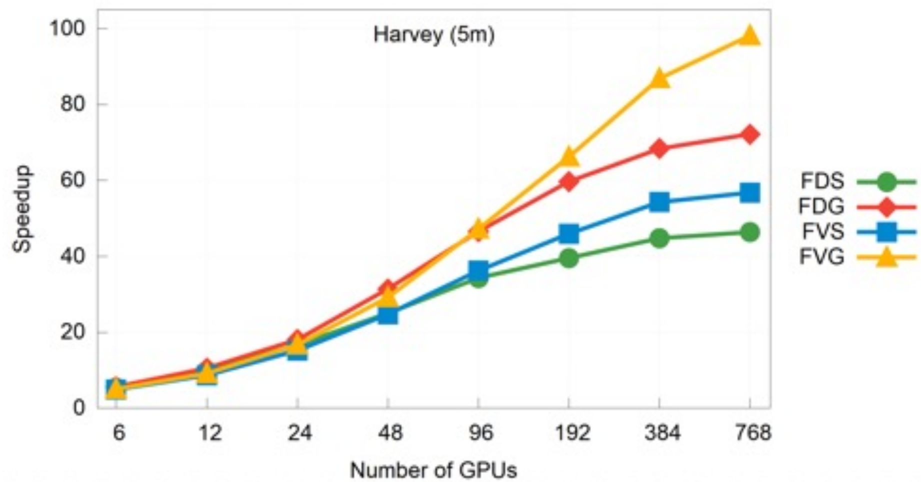
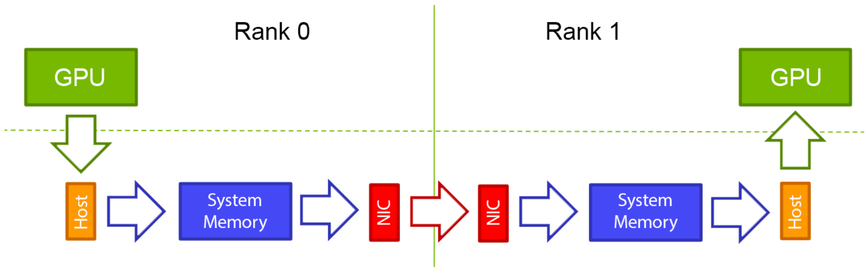
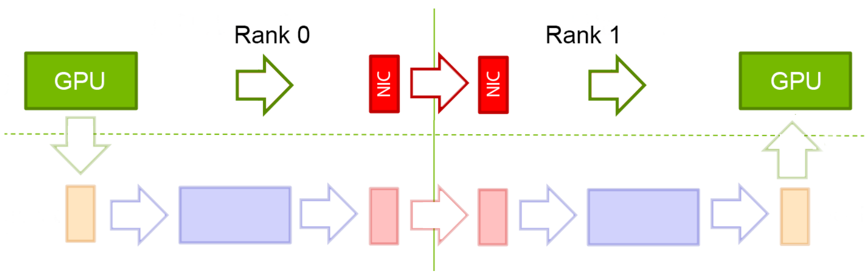


Fig: Speed Up

TRITON (3/4)



Standard



GPU-Direct

Fig: GPU to GPU Communication

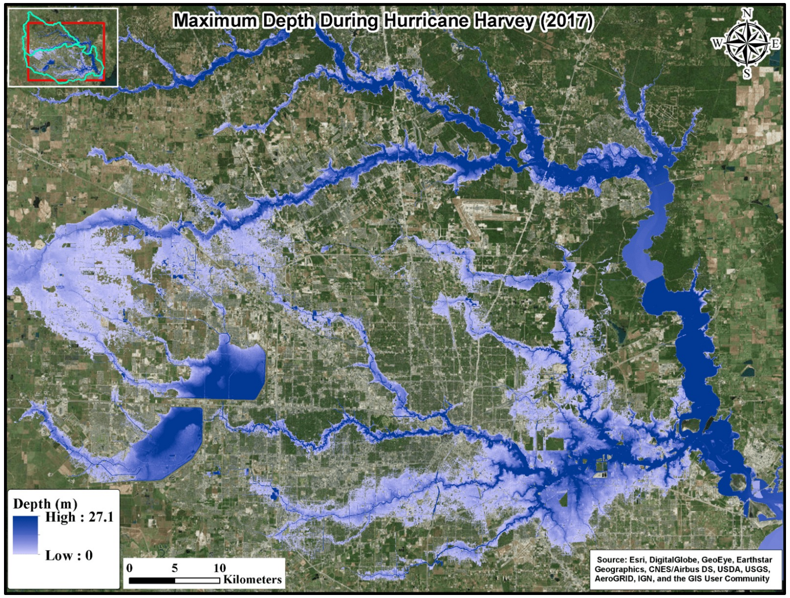


Fig: Max depth during Hurricane Harvey

TRITON (4/4)

- Contribution
  - Development of TRITON
  - Open Source
  - Performance Analysis
  - Different model implementation
  - Efficient, Scalable, Multi GPU implementation
    - 128 nodes (768 GPUs)
  - 10 day simulation of Hurricane Harvey finished in 50 minutes (272 million cells)
    - Serial version: Months? Years?
- Publications (5)
  - Performance Evaluation of a Two-Dimensional Flood Model (Conference)
  - TRITON:A Multi-GPU open source 2D hydrodynamic flood model (Journal)
  - High-performance computing in water resources (Journal)
  - Simulation of Hurricane Harvey flood event (Journal)
  - Assessing climate-change-induced flood risk in the Conasauga River watershed (Journal)

Components

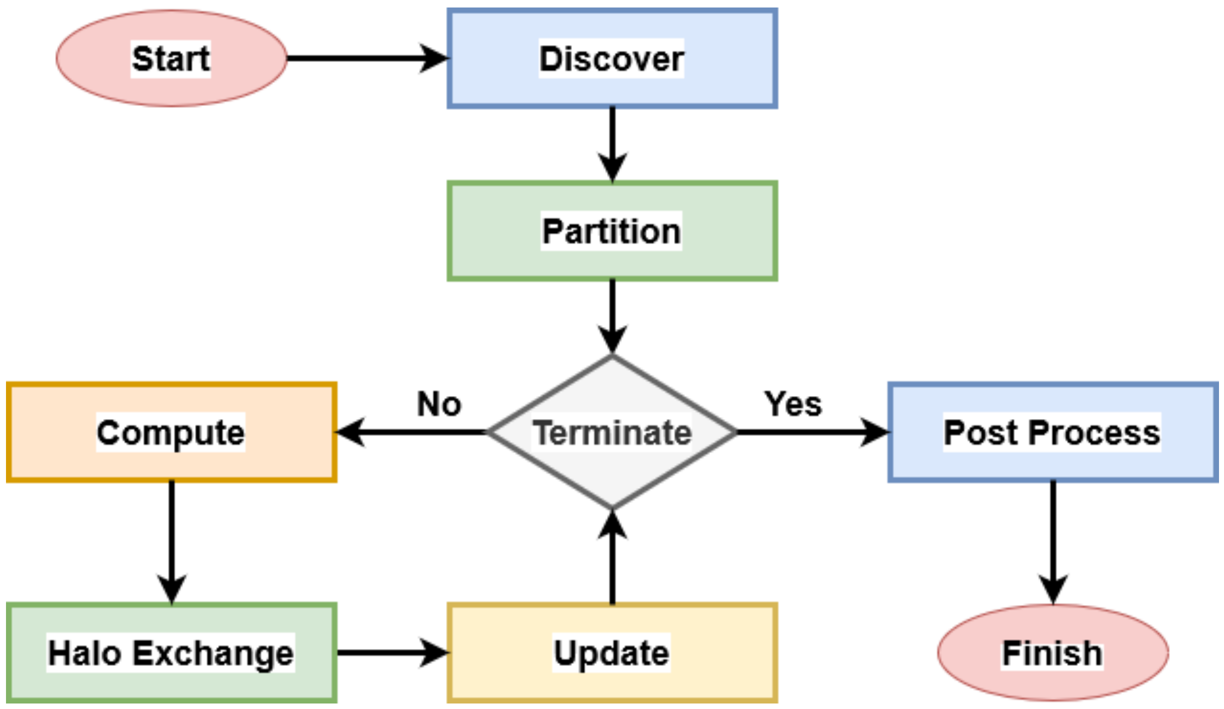


Fig: Different components of stencil application flow diagram in APPFIS

Framework Design

- APPFIS: **A**dvanced **P**arallel **P**rogramming **F**ramework for **I**terative **S**tencil

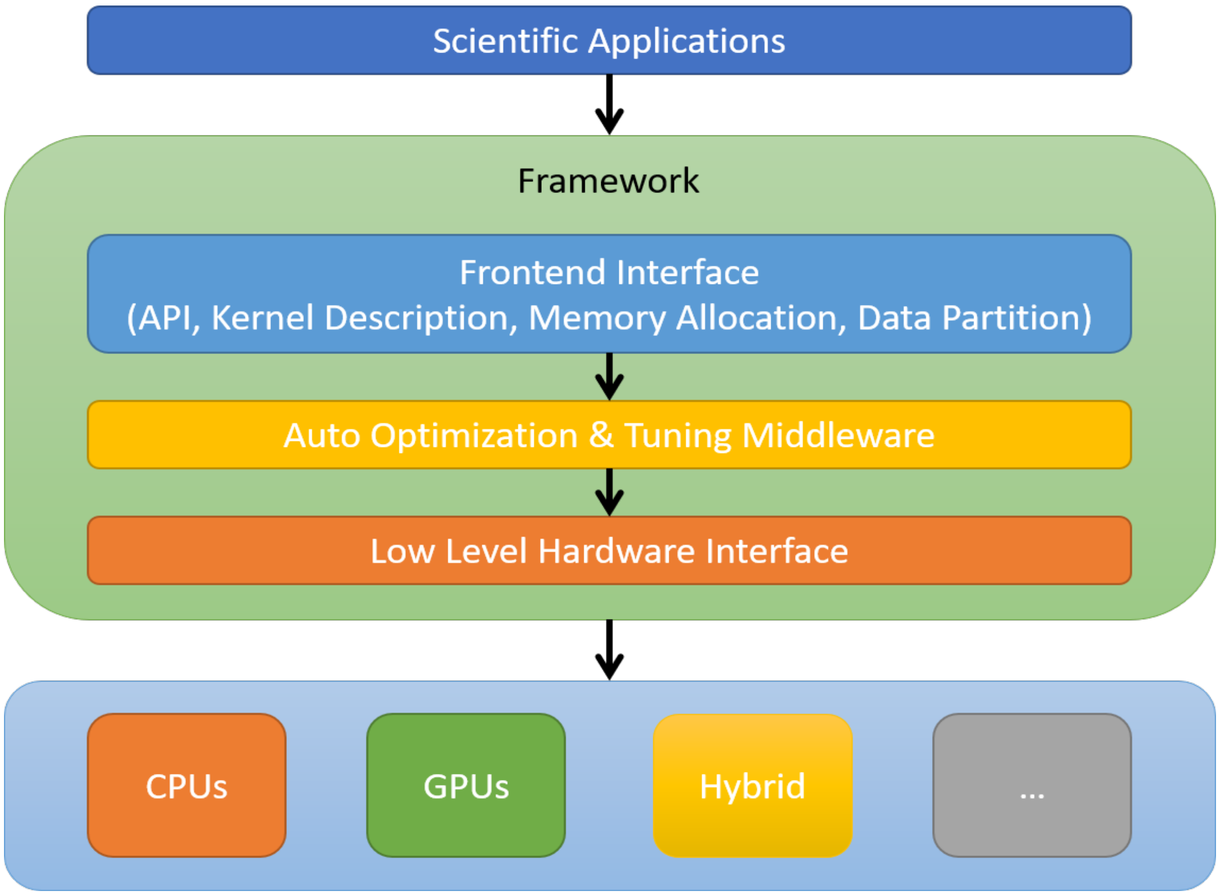


Fig: High-level software development model using APPFIS framework

Optimization

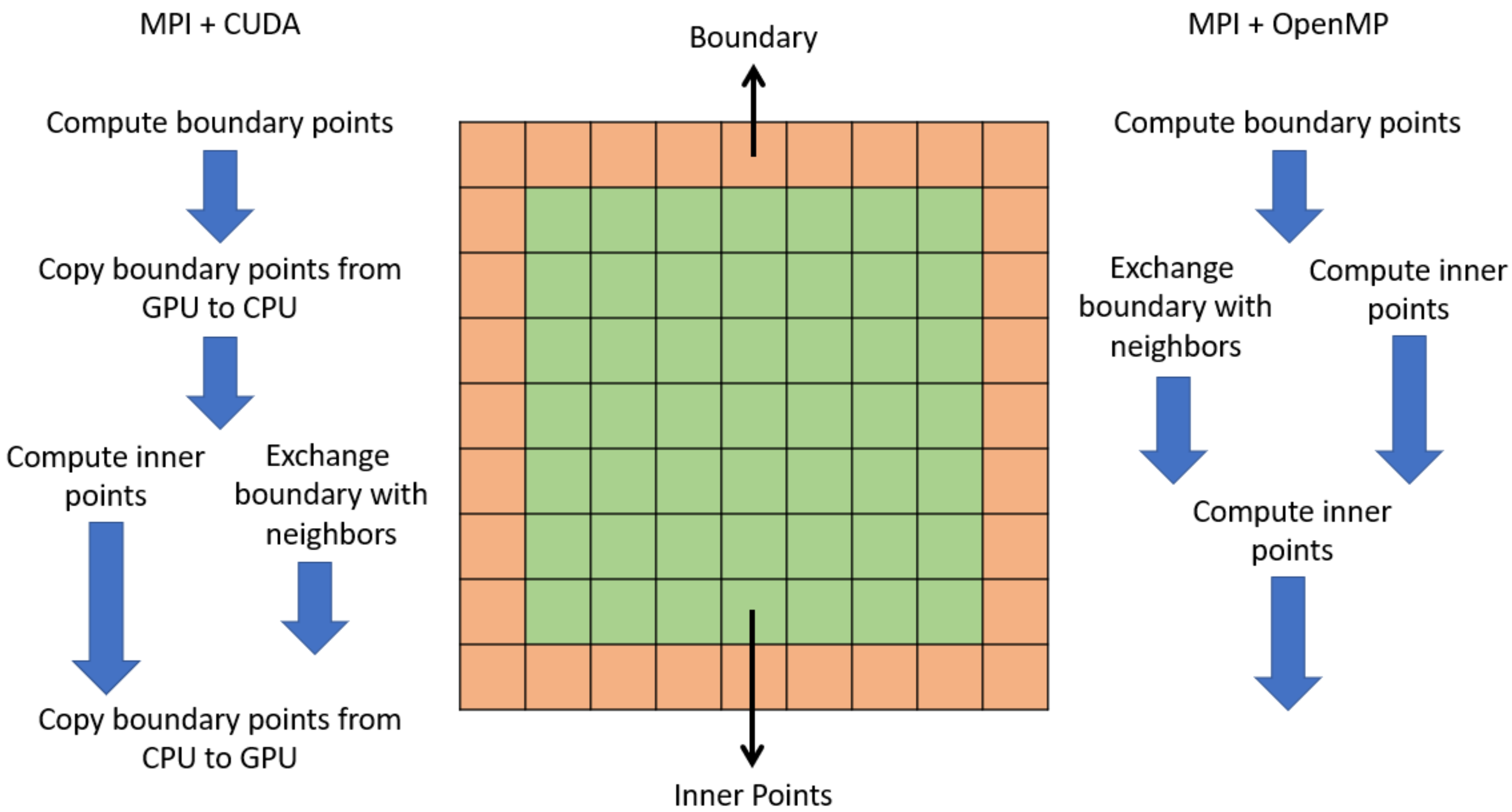


Fig: Computation Communication overlapping

Programming Model

- Initialization and Finalization
  - **void** Initialize(**int** argc, **char*** argv[], **int** dataDim, **int** partitionDim);
  - **void** Finalize();
- *Grid and Array*
  - Grid<T>(**int** nDim, **int*** dims, GRID_TYPE t);
  - Array<T>(**int** nDim, **int*** dims);
- Scatter and Gather
  - **void** Scatter(Array<T>* a, Array<T>* sub);
  - **void** Gather(Array<T>* a, Array<T>* sub);
- Stencil
  - ```
    void STENCIL(const int i, const int j, Array<int>* cur, Array<int>* next) {
        int val = cur->Get(i, j) + cur->Get(i - 1, j) + cur->Get(i + 1, j) + cur->Get(i, j – 1) +
                  cur->Get(i, j + 1);
        next->Set(i, j, val);
    }
    ```
    ```

Programming Model

- Execute
  - Execute2D(Config<**int**> c, Array<**int**>* cur, Array<**int**>* next);
- Reduction
  - REDUCTION, Grid<T> g);

```
void LoadAsciiFile(string path);
void SaveBinaryFile(string path);
```

```
struct Attribute {
        Attribute() : ghostLayer(1),    threads(1), periodic(false),    overlap(false) {}
        int ghostLayer;
        int threads;
        bool periodic;
        bool overlap;
} ATTRIBUTE;
```

Application and Architecture

- Application
  - TRITON
  - Game of Life
  - Heat Diffusion
  - Wave Equation
  - Himeno

- Architecture
  - Stampede2 at Texas Advanced Computing Center
  - Bridges-2 at Pittsburgh Supercomputing Center
  - Summit Supercomputer at Oak Ridge National Laboratory
  - HPC Cluster at Tennessee Tech University

- 4096 CPUs and 384 GPUs

Scalability

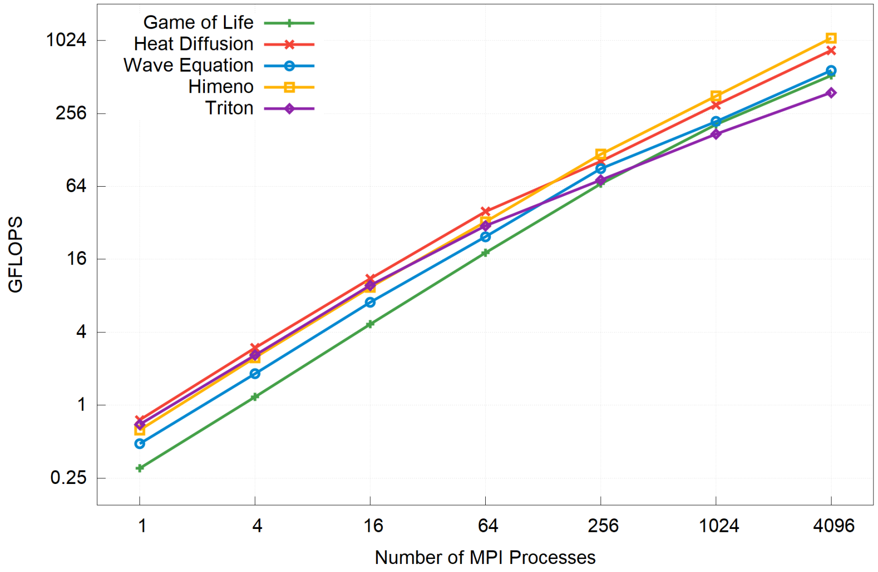


Fig: Strong (Stampede2)

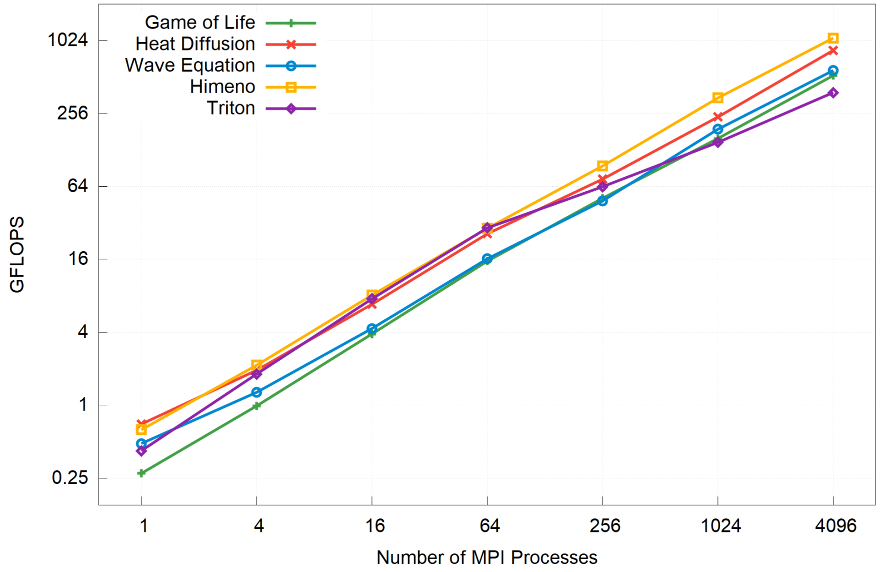


Fig: Weak (Stampede2)

Scalability

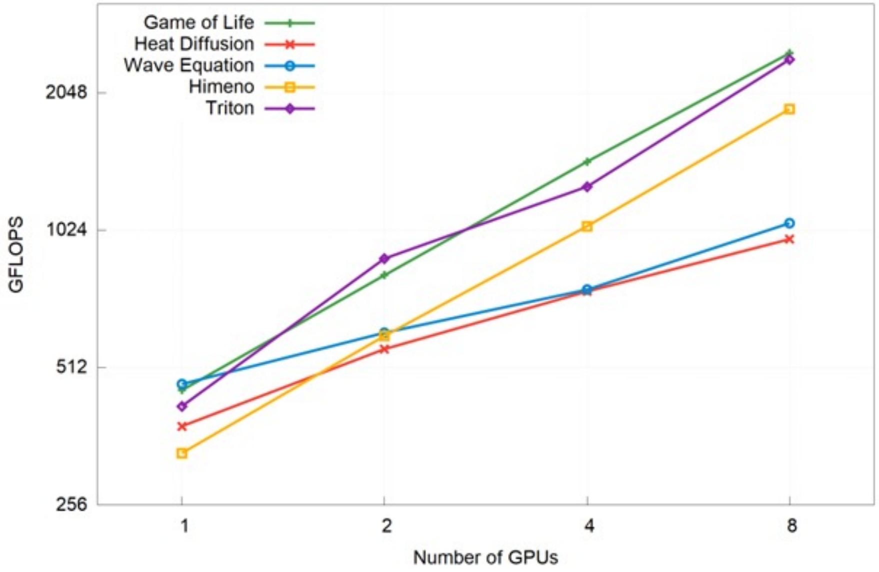


Fig: Strong (Bridges2)

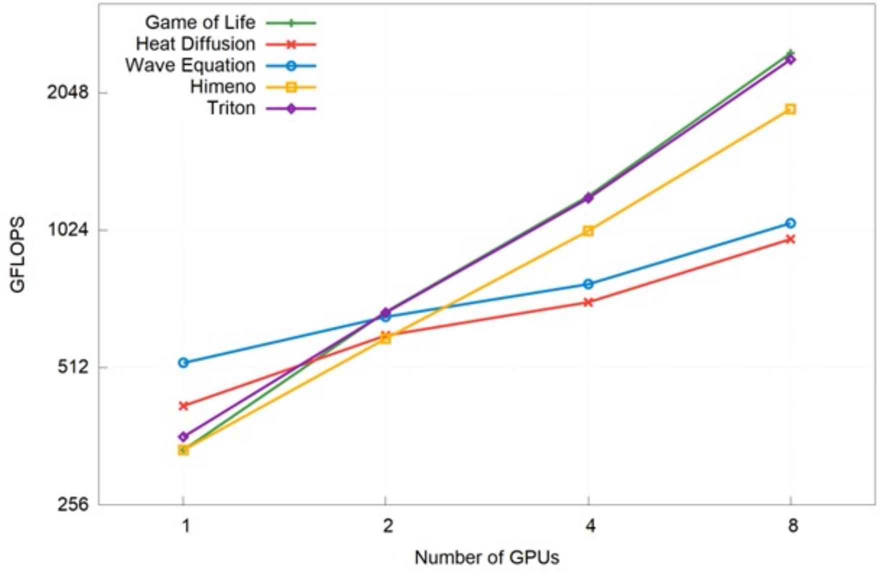


Fig: Weak (Bridges2)

APPFIS vs Original

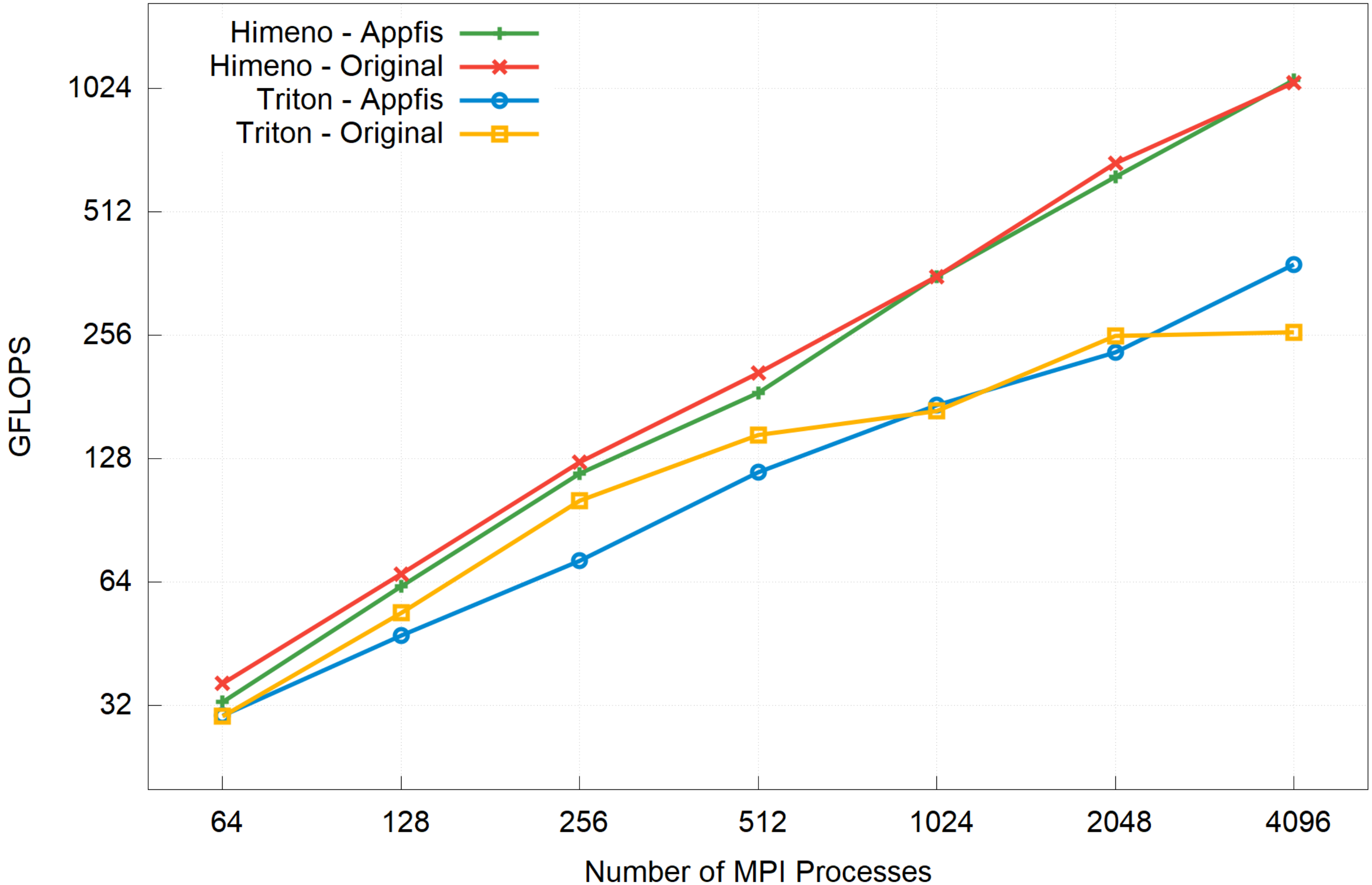


Fig: APPFIS vs Original (Stampede2)

TRITON Performance

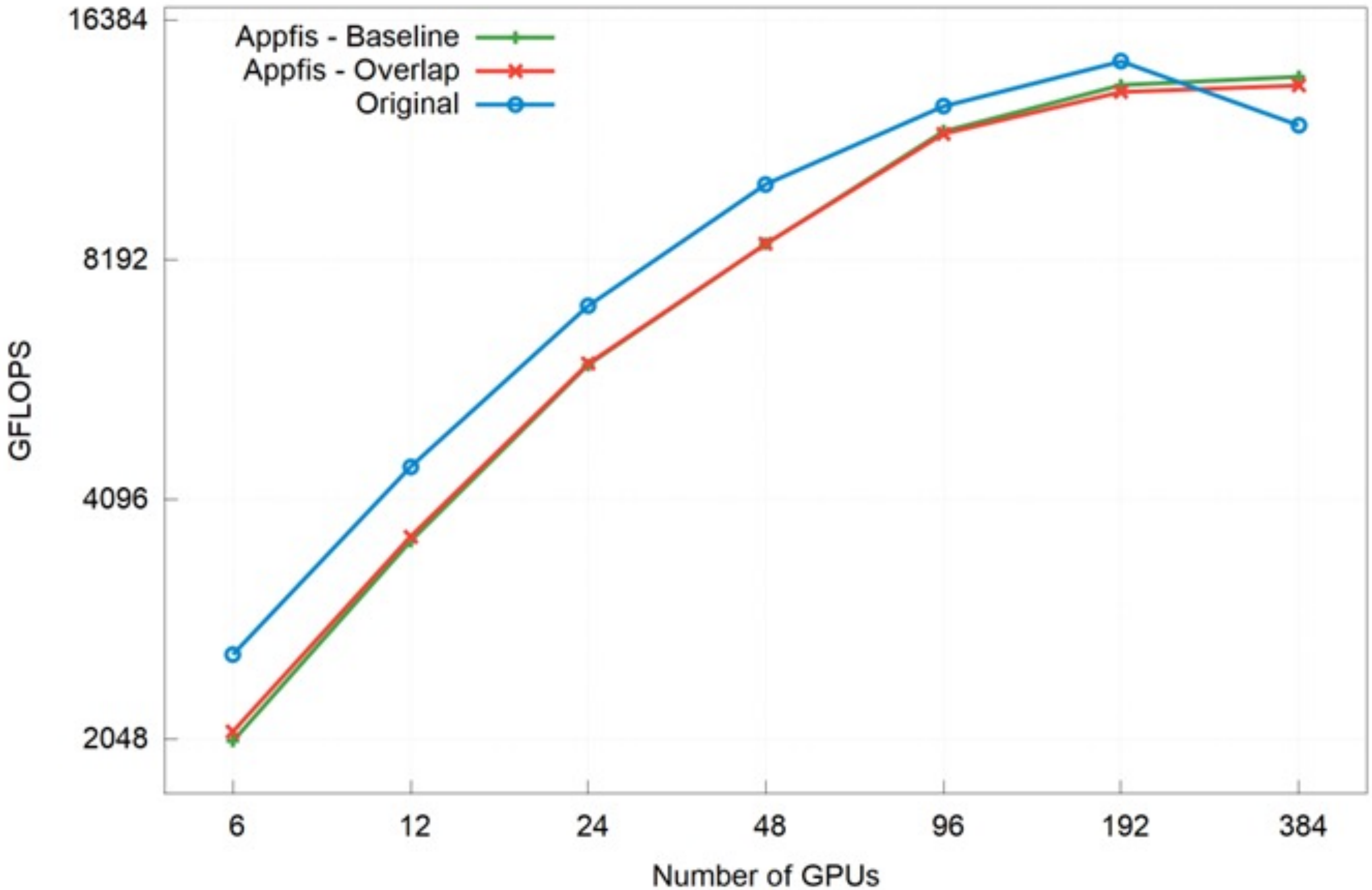


Fig: TRITON (Summit)

Optimization

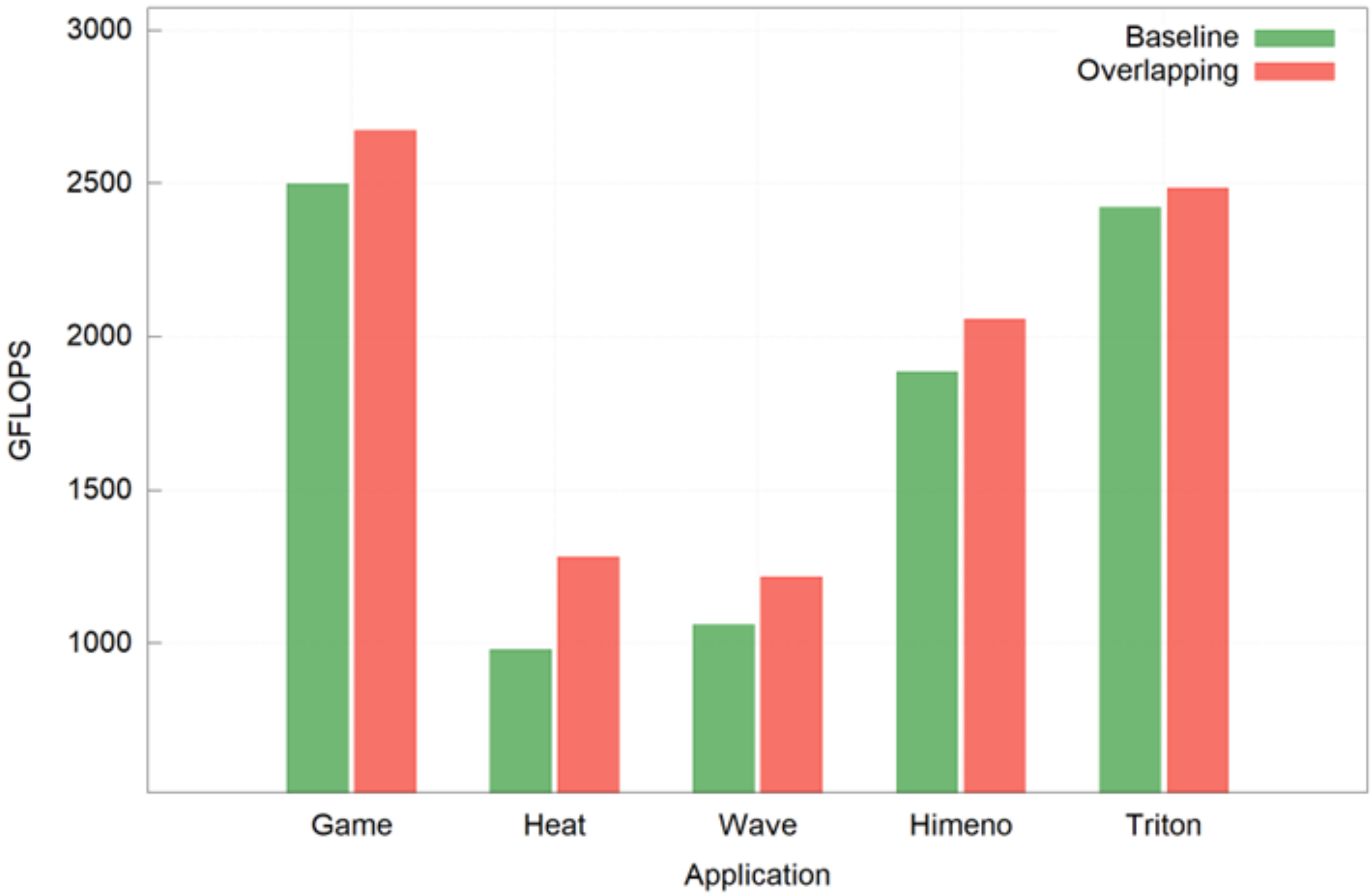


Fig: Effect of overlapping (Bridges2)

Conclusion

- Designed and developed APPFIS
- Open Source (https://github.com/msharif42/APPFIS)
- Easy to use interface
- Shared Memory, Distributed Memory, Nvidia GPUs
- Auto Partition (Multi Dimension)
- Auto memory management
- Optimization
- Hardware agnostic frontend
- Automatic communication
- Good strong and weak scalability
- Performance loss at most 20% compare to hand tuned code
- FPGA, Different GPU, Future Architecture
    - APPFIS developer will include hardware dependent implementation
    - Application developer need not to change any code

- **Acknowledgement: US Air Force, Oak Ridge National Lab, NSF XCEDE Project**

References

1. Maruyama, Naoya, et al. "Physis: an implicitly parallel programming model for stencil computations on large-scale GPU-accelerated supercomputers." *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*. 2011.
2. Lutz, Thibaut, Christian Fensch, and Murray Cole. "PARTANS: An autotuning framework for stencil computation on multi-GPU systems." *ACM Transactions on Architecture and Code Optimization (TACO)* 9.4 (2013): 1-24.
3. Christen, Matthias, Olaf Schenk, and Helmar Burkhart. "Patus: A code generation and autotuning framework for parallel iterative stencil computations on modern microarchitectures." *2011 IEEE International Parallel & Distributed Processing Symposium*. IEEE, 2011.
4. Sourouri, Mohammed, Scott B. Baden, and Xing Cai. "Panda: A Compiler Framework for Concurrent CPU-GPU Execution of 3D Stencil Computations on GPU-accelerated Supercomputers." *International Journal of Parallel Programming* 45.3 (2017): 711-729.
5. Gysi, Tobias, et al. "STELLA: A domain-specific tool for structured grid methods in weather and climate models." *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. 2015.
6. Pereira, Alyson D., Luiz Ramos, and Luís FW Góes. "PSkel: A stencil programming framework for CPU-GPU systems." *Concurrency and Computation: Practice and Experience* 27.17 (2015): 4938-4953.
7. Edwards, H. Carter, Christian R. Trott, and Daniel Sunderland. "Kokkos: Enabling manycore performance portability through polymorphic memory access patterns." *Journal of Parallel and Distributed Computing* 74.12 (2014): 3202-3216.
8. Hornung, Richard D., and Jeffrey A. Keasler. *The RAJA portability layer: overview and status*. No. LLNL-TR-661403. Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States), 2014.

Thank You

Any Question?