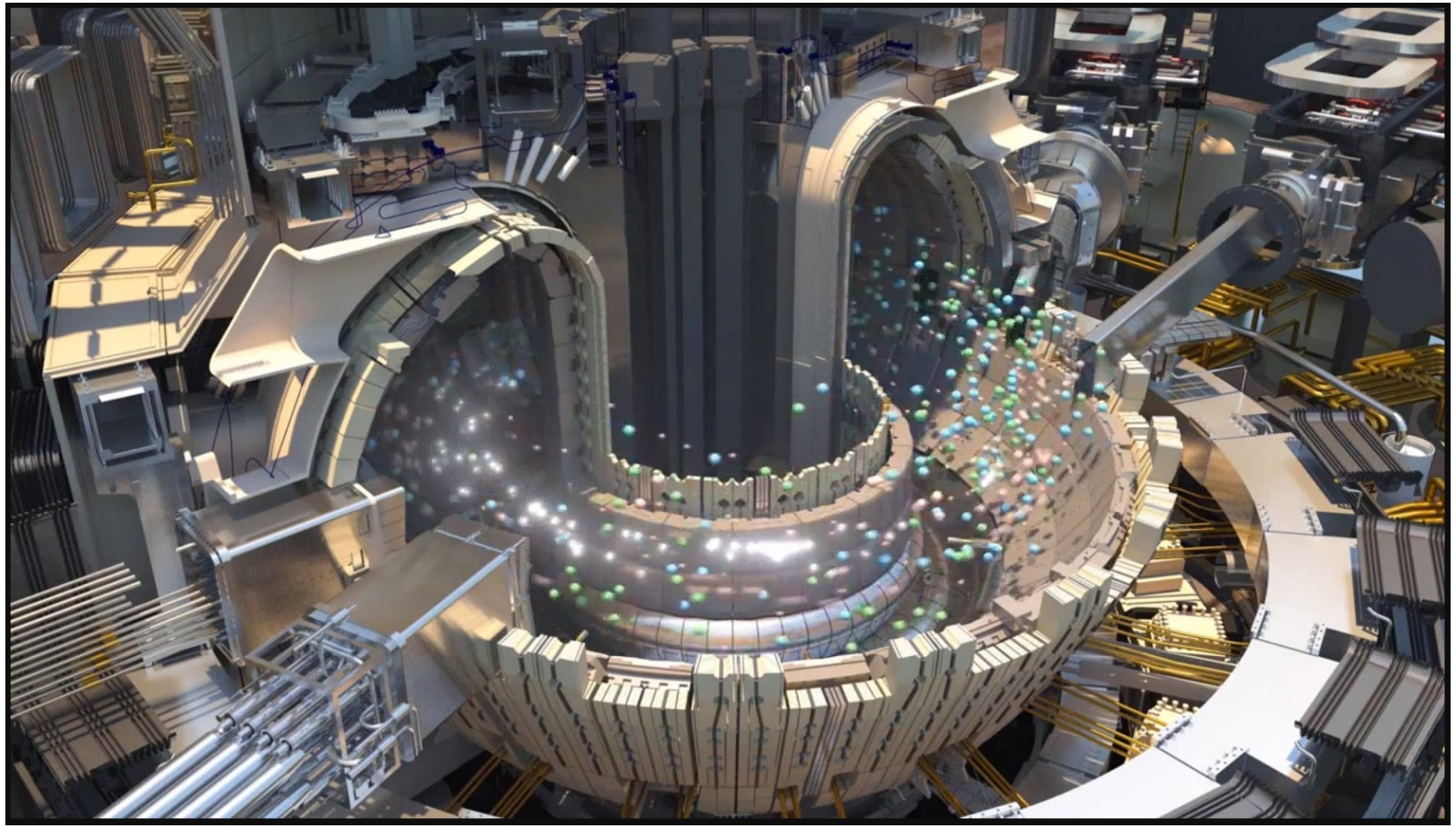
NESAP Application: WDMApp

Dhruva Kulkarni
Application Performance Group (APG)

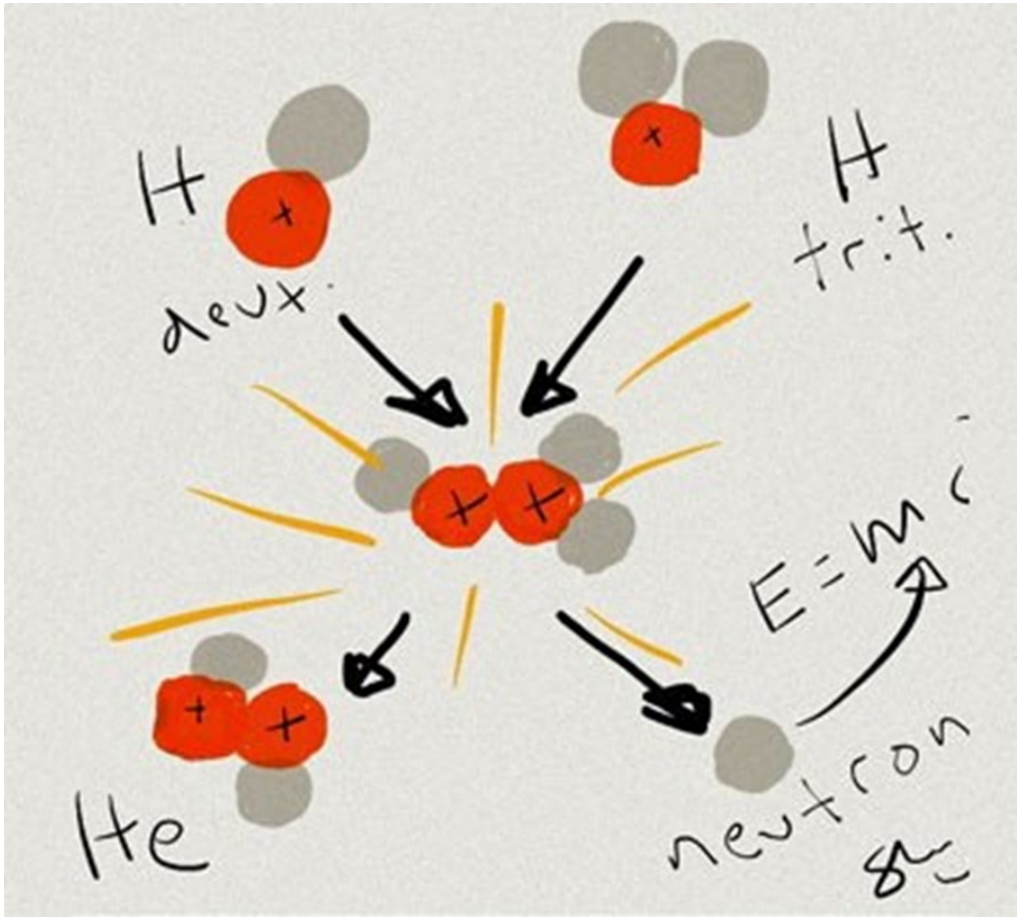
Current Mission: Make codes run faster on GPUs!

Overview:
Introduce WDMApp at a very general level
Describe some performance optimizations
Present some optimization results

What is this a picture of?

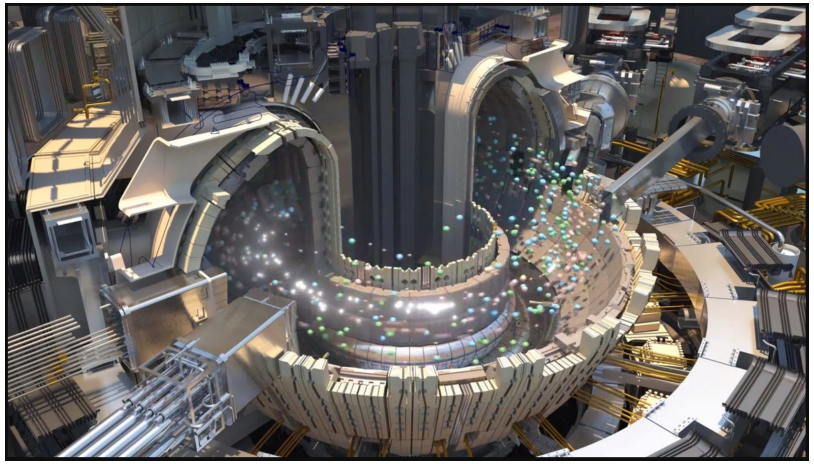


ITER Fusion Reactor



www.iter.org

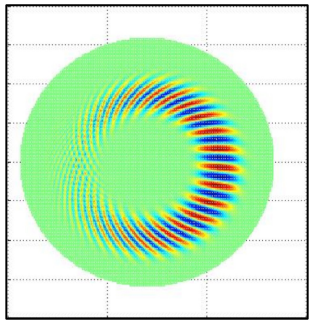
Magnetically confined fusion
"Tokamak" – Russian acronym for
"toroidal chamber with magnetic coils"
Converts the heat of fusion into useful forms (steam to electricity via turbines)



Whole Device Modeling Application (WDMApp)

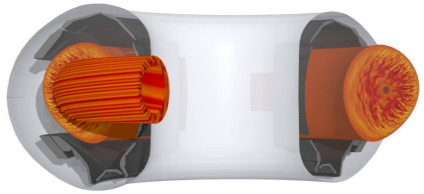
- ITER – a never before accessed experimental regime
- Need for hi-fidelity whole device simulation to drive design choices
- Princeton Plasma Physics Lab

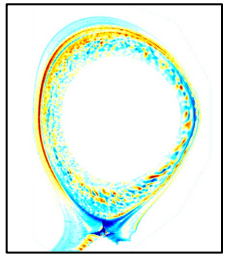
WDMApp Requires Exascale Computers and Beyond



Gigaflops

5-D electrostatic ion physics in simplified circular cylindrical geometry

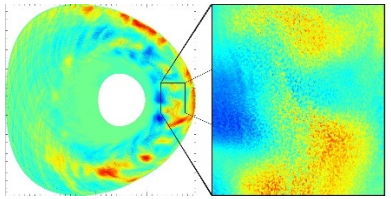


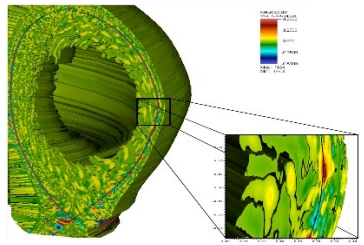


Teraflops

Core: 5D ion-scale electromagnetic physics in torus

Edge: ion+neutral electrostatic physics in torus

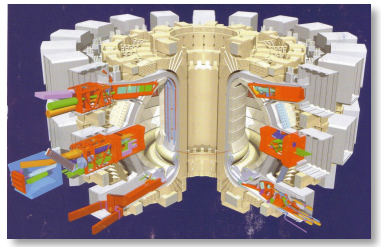




Petaflops

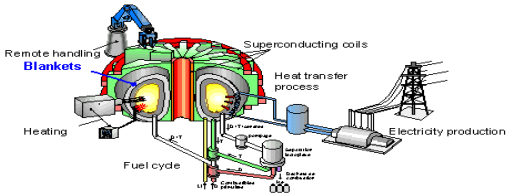
Core: 5D ion scale Maxwellian ion + electron electromagnetic

Edge: non-Maxwellian plasma, electrostatic physics



Exaflops

Core-edge coupled 5D electromagnetic study of whole-device ITER, including ion-scale turbulence + local electron-scale turbulence, profile evolution, large-scale instability, plasma-material interaction, rf heating, and energetic particles

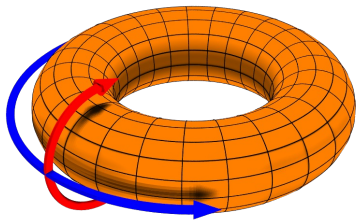


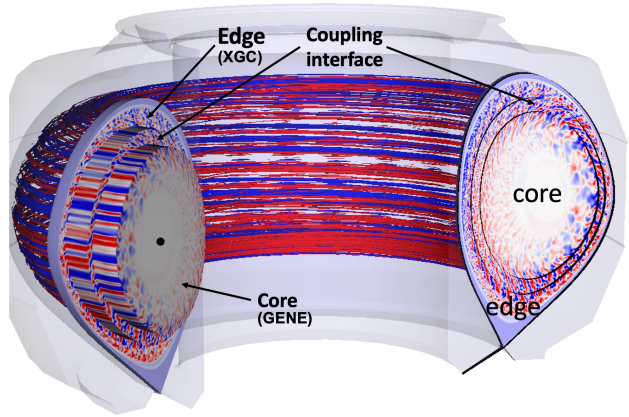
Beyond

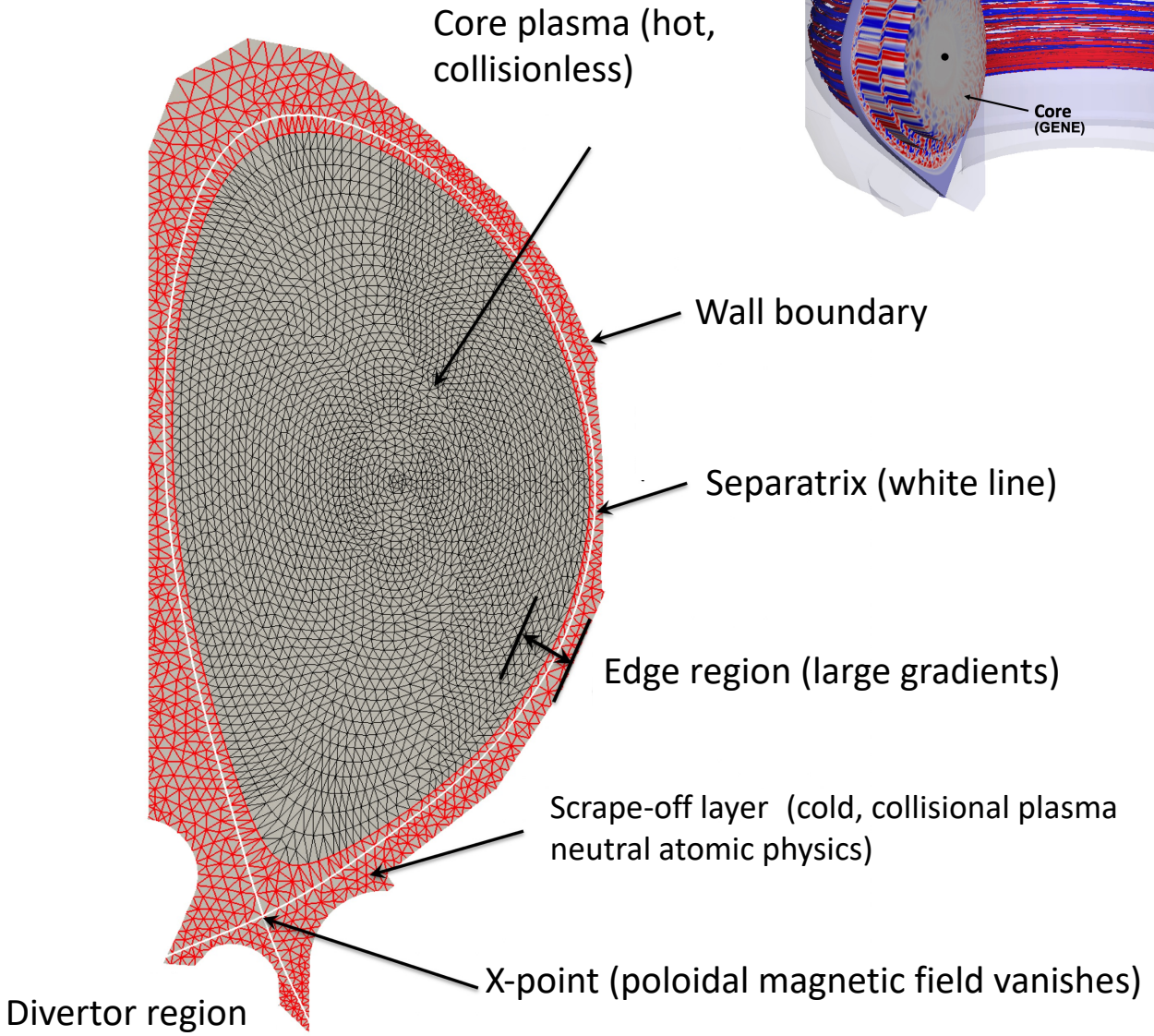
A WDMApp that includes necessary engineering reactor components, and applicable to leading alternate concepts (including stellarators); and possibly 6D whole device modeling

XGC

- X-Point included Gyrokinetic Code
- Gyrokinetic PIC
  - Multi-species
  - Drift-kinetic electrons
- Unstructured mesh in poloidal plane
  - ITER 3mm resolution: 900K nodes
  - Same every plane
  - Small number of planes (32-128)
  - Mapping particles to cells is complex



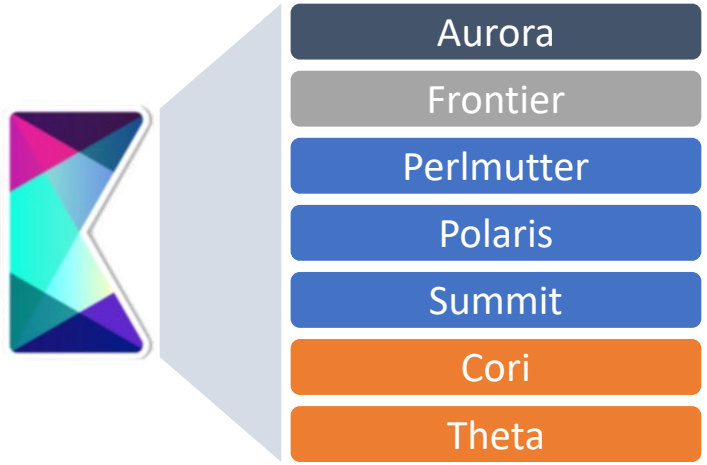


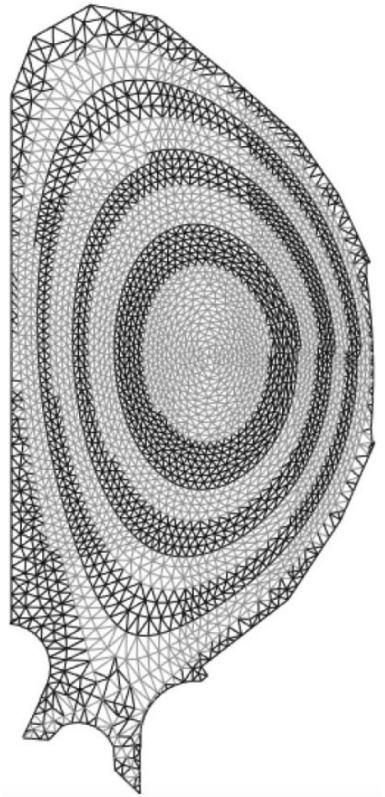


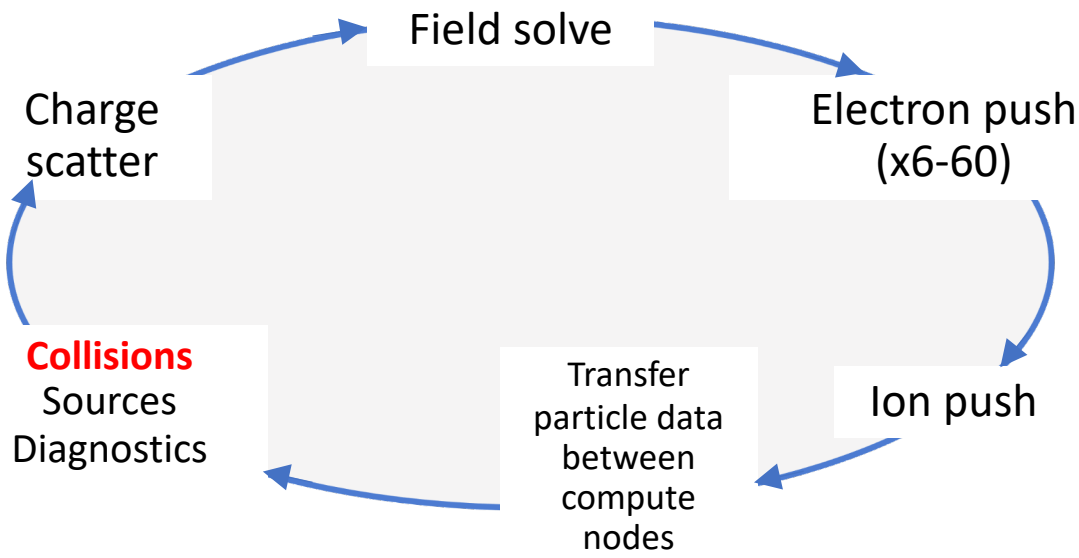
Figures: E. D'Azevedo et al., in Exascale Scientific Applications: Scalability and Performance Portability. Chapman & Hall/CRC 2017

Parallelism

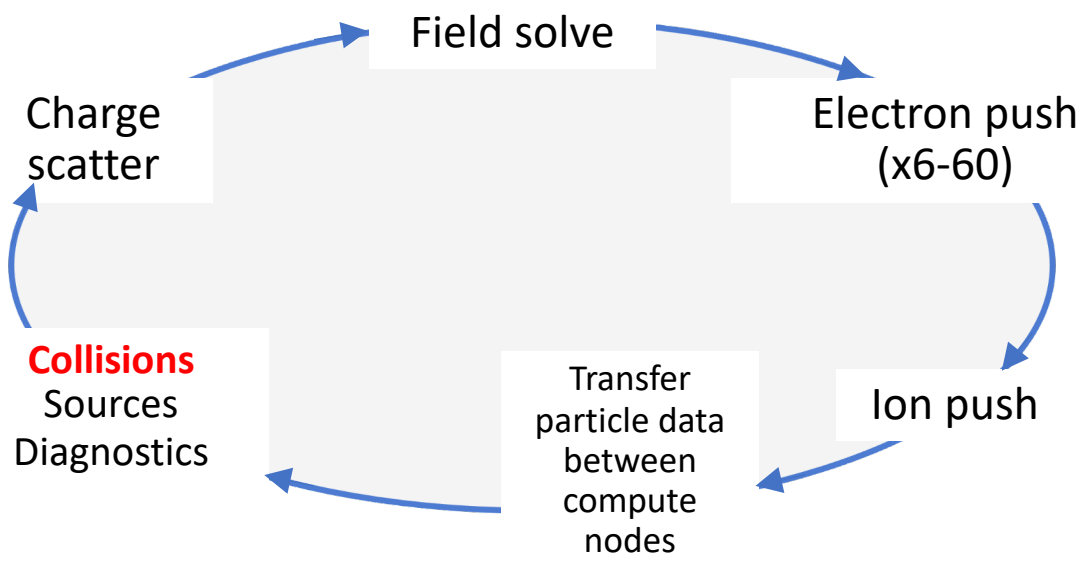
- MPI
  - Toroidal decomposition
  - Spiral decomposition in cut planes
- Intranode parallelism
  - CPU
    - OpenMP threads
    - Vectorization (via Kokkos)
  - GPU
    - Kokkos is our portability layer
      - Cabana library from ECP CoPA project (built on Kokkos)

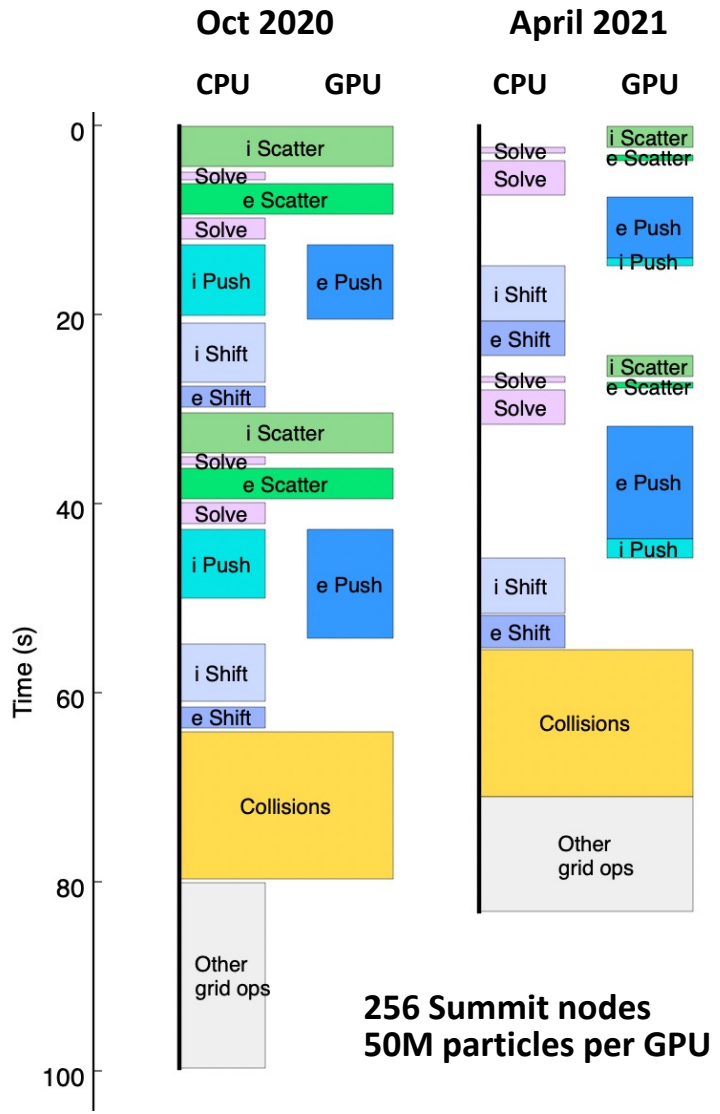






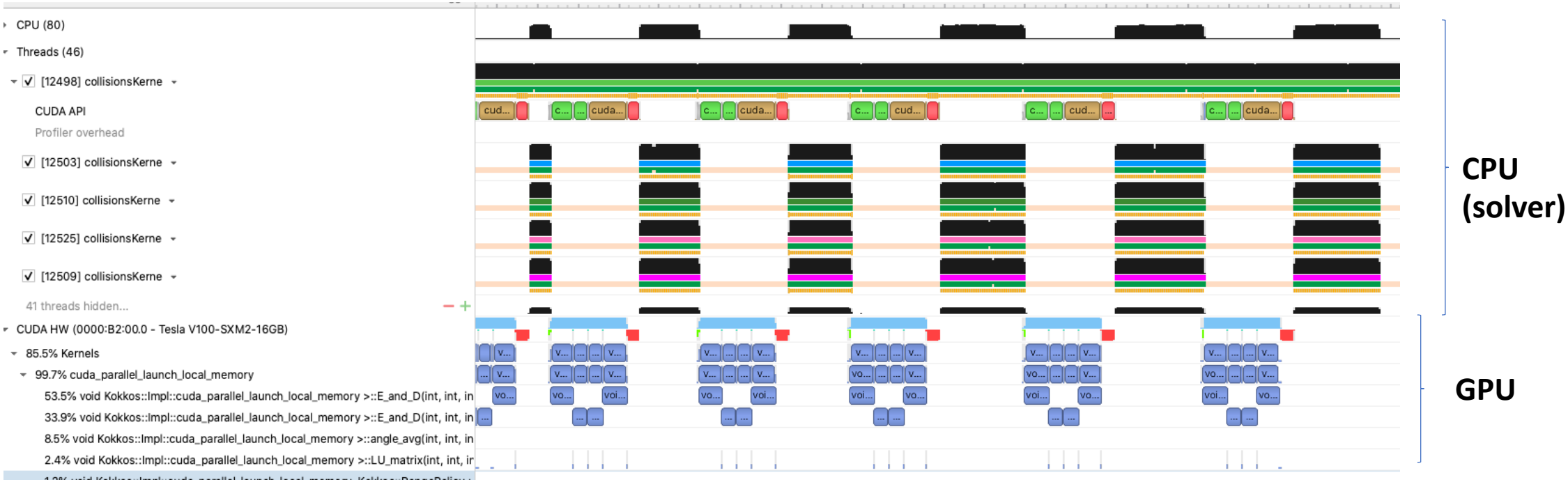
XGC Perf. Optimization – Overview





XGC Collision Kernel

Profiler: “Nsight systems”



- Profile, eat, optimize, eat, sleep, repeat…..
- i. Kokkos ←→ OpenMP Conversion and lessons learnt therefrom; ii. Porting a linear Solver to GPUs
- Publications:

IWOMP 2021: Outcomes of OpenMP Hackathon: OpenMP Application Experiences with the Offloading Model (Part I)

Accepted in IPDPS 2022: Batched sparse iterative solvers on GPU for the collision operator for fusion plasma simulations

Follow on paper invited: JPDC 2022

Base OpenMP version (translated from Kokkos)

```
Kokkos::parallel_for(”E_and_D_s",
Opt::require(policy(col_f_nvrm1_nvzm1),Async),
KOKKOS_LAMBDA (const int idx) {
    E_and_D_s(…);
});
```

```
#pragma omp target teams distribute parallel for
is_device_ptr(...) map(...)
for (int idx=0; idx<col_f_nvrm1_nvzm1; idx++)
  E_and_D_s_omptarget(...);
```

Coarse-grained parallelism only: every thread executes the E_and_D_s_omptarget function.

Cannot perform well on the GPU without streams

	Kokkos	Base OpenMP
ED_ab	0.94	3.16
ED_s	0.59	2.56

Time in seconds
V100, NVIDIA HPC SDK 20.11.
Problem used 100 mesh nodes

Base OpenMP version (translated from Kokkos)

```
Kokkos::parallel_for(”E_and_D_s",
Opt::require(policy(col_f_nvrm1_nvzm1),
Async),KOKKOS_LAMBDA (const int idx) {
    E_and_D_s(…);
});
```

```
#pragma omp target teams distribute parallel for
is_device_ptr(...) map(...)
for (int idx=0; idx<col_f_nvrm1_nvzm1; idx++)
  E_and_D_s_omptarget(...);
```

```
void E_and_D_s_omptarget(...) {
  for (index_ip = 0; index_ip<nvzm1; index_ip++) {
    for (index_jp = 0; index_jp<nvrm1; index_jp++) {
      // ...
```

OpenMP Hackathon:
Use fine grained parallelism
(Parallelize inner loops)

OpenMP code – Adding inner parallelism

```
#pragma omp target data map(...)
  E_and_D_s_omptarget_v2(...);

void E_and_D_s_omptarget_v2(...) {
#pragma omp target teams distribute is_device_ptr(...)
  for (int idx=0; idx<col_f_nvrm1_nvzm1; idx++) {
    // ...
#pragma omp parallel for collapse(2) reduction(+:...)
    for (index_ip = 0; index_ip<nvzm1; index_ip++) {
      for (index_jp = 0; index_jp<nvrm1; index_jp++) {
```

Outer loop moved inside the function
Inner 2 loops parallelized and workshared over threads

	Kokkos	Base OpenMP	Fine-grained OpenMP
ED_ab	0.94	3.16	1.09
ED_s	0.59	2.56	0.22

Time in seconds
V100, NVIDIA HPC SDK 20.11.
Problem used 100 mesh nodes

OpenMP code – loop construct

```
#pragma omp target data map(...)
  E_and_D_s_omptarget_v2(...);

void E_and_D_s_omptarget_v2(...) {
#pragma omp target teams loop is_device_ptr(...)
  for (int idx=0; idx<col_f_nvrm1_nvzm1; idx++) {
    // ...
#pragma omp loop collapse(2) reduction(+:...)
    for (index_ip = 0; index_ip<nvzm1; index_ip++) {
      for (index_jp = 0; index_jp<nvrm1; index_jp++) {
```

OpenMP-4.5 compute constructs replaced with OpenMP-5.0 “loop” construct

Initial runs with “loop” construct were very slow

Time in seconds
V100, NVIDIA HPC SDK 20.11.
Problem used 100 mesh nodes

OpenMP code – loop construct

```
#pragma omp target data map(...)
  E_and_D_s_omptarget_v2(...);
#define COL_F_NVRM1_NVZM1 930
void E_and_D_s_omptarget_v2(...) {
#pragma omp target teams loop is_device_ptr(...)
  for (int idx=0; idx<COL_F_NVRM1_NVZM1; idx++) {
    // ...
#pragma omp loop collapse(2) reduction(+:...)
    for (index_ip = 0; index_ip<nvzm1; index_ip++) {
      for (index_jp = 0; index_jp<nvrm1; index_jp++) {
```

Make outer loop upper bound a compile time constant

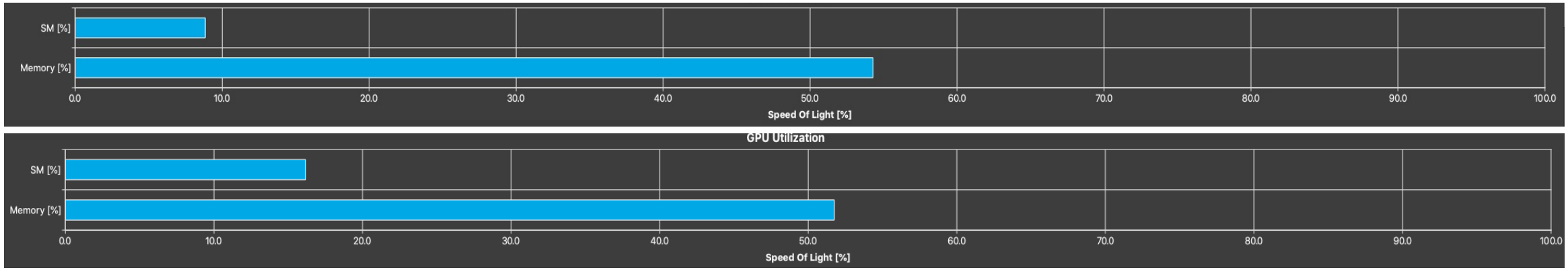
Workaround to get more OpenMP teams. Should be fixed in HPC SDK 21.3

	Kokkos	Base OpenMP	Fine-grained OpenMP	Fine-grained w/ "loop"
ED_ab	0.94	3.16	1.09	0.21

Time in seconds
V100, NVIDIA HPC SDK 20.11.
Problem used 100 mesh nodes

A few hackathons later

Profiler: “Nsight compute”



Fine-grained

Loop

Time in seconds
V100, NVIDIA HPC SDK 20.11. Problem used 100 mesh nodes

Kokkos (+ Cuda streams)
OpenMP – parallelizing nested loops
Kokkos – Batching

Working closely with NVIDIA compiler engineers

	Kokkos	Base OpenMP	Fine-grained OpenMP	Fine-grained w/ “loop”
ED_ab	0.94	3.16	1.09	0.21
ED_s	0.59	2.56	0.22	0.16

NERSC

OAK RIDGE National Laboratory

XGC Collision Kernel

Profiler: “Nsight systems”

- Profile, eat, optimize, eat, sleep, repeat…..
- i. Kokkos ←→ OpenMP Conversion and lessons learnt therefrom; ii. Porting a linear Solver to GPUs
- Publications:

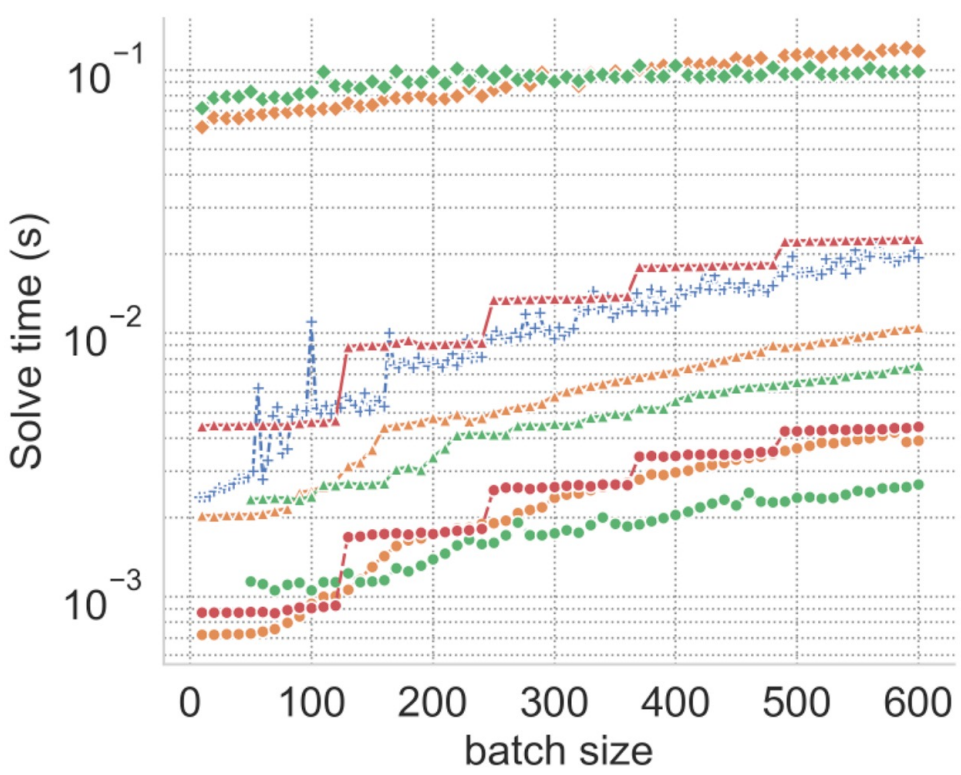
IWOMP 2021: Outcomes of OpenMP Hackathon: OpenMP Application Experiences with the Offloading Model (Part I)

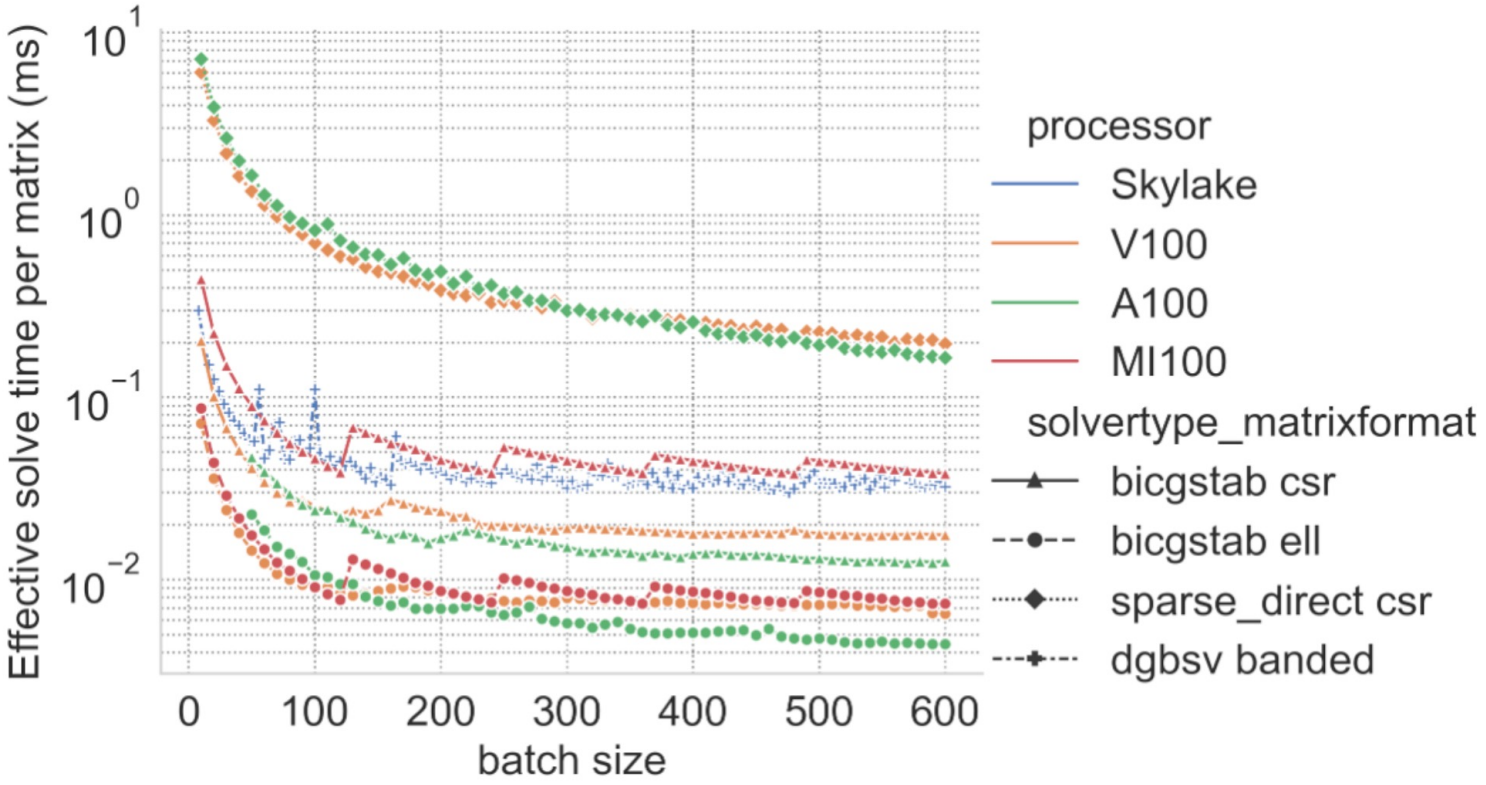
Accepted in IPDPS 2022: Batched sparse iterative solvers on GPU for the collision operator for fusion plasma simulations

Follow on paper invited: JPDC 2022

XGC Collision Kernel and ECP Ginkgo

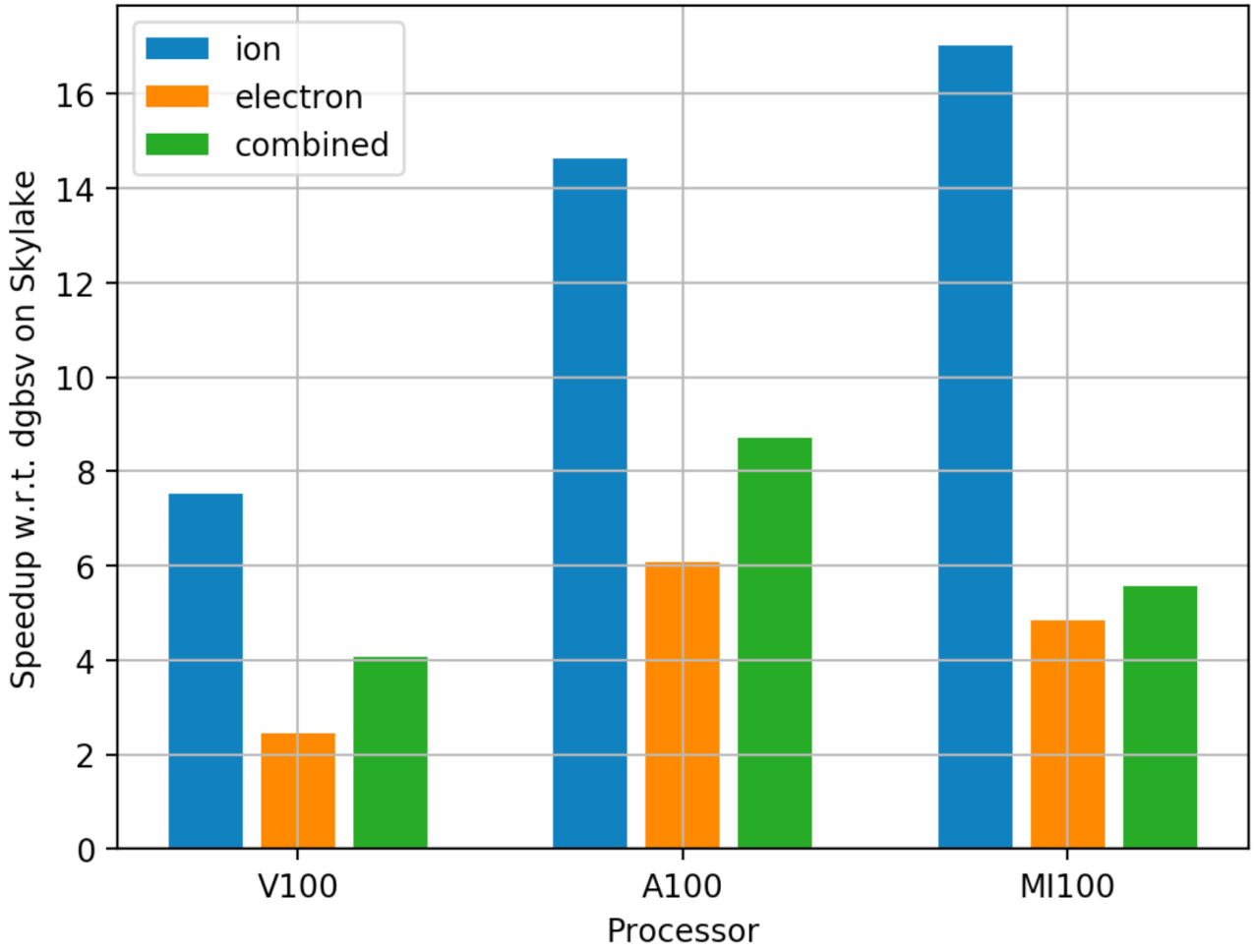
Comparisons on various architectures for different solvers and matrix formats



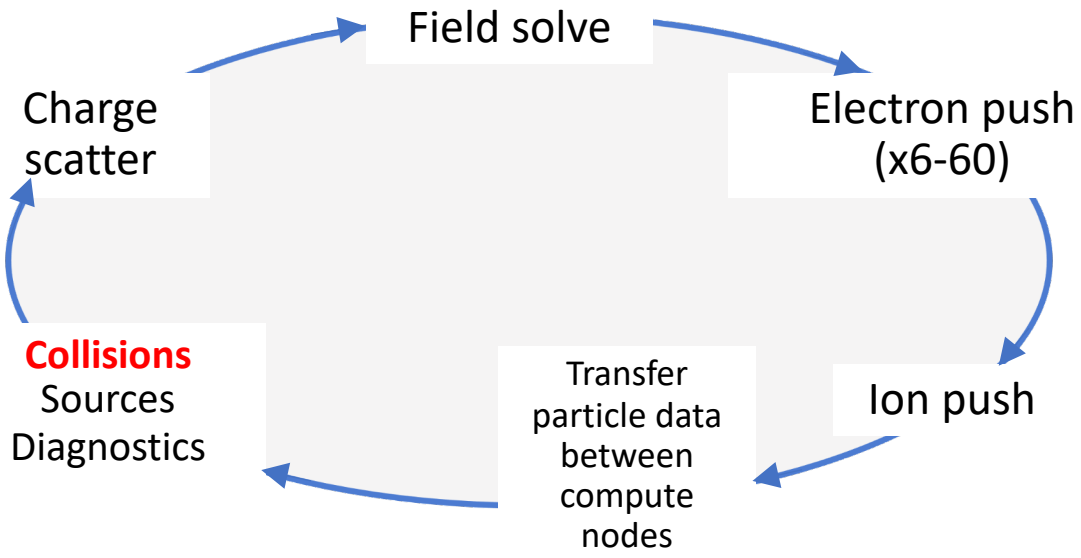


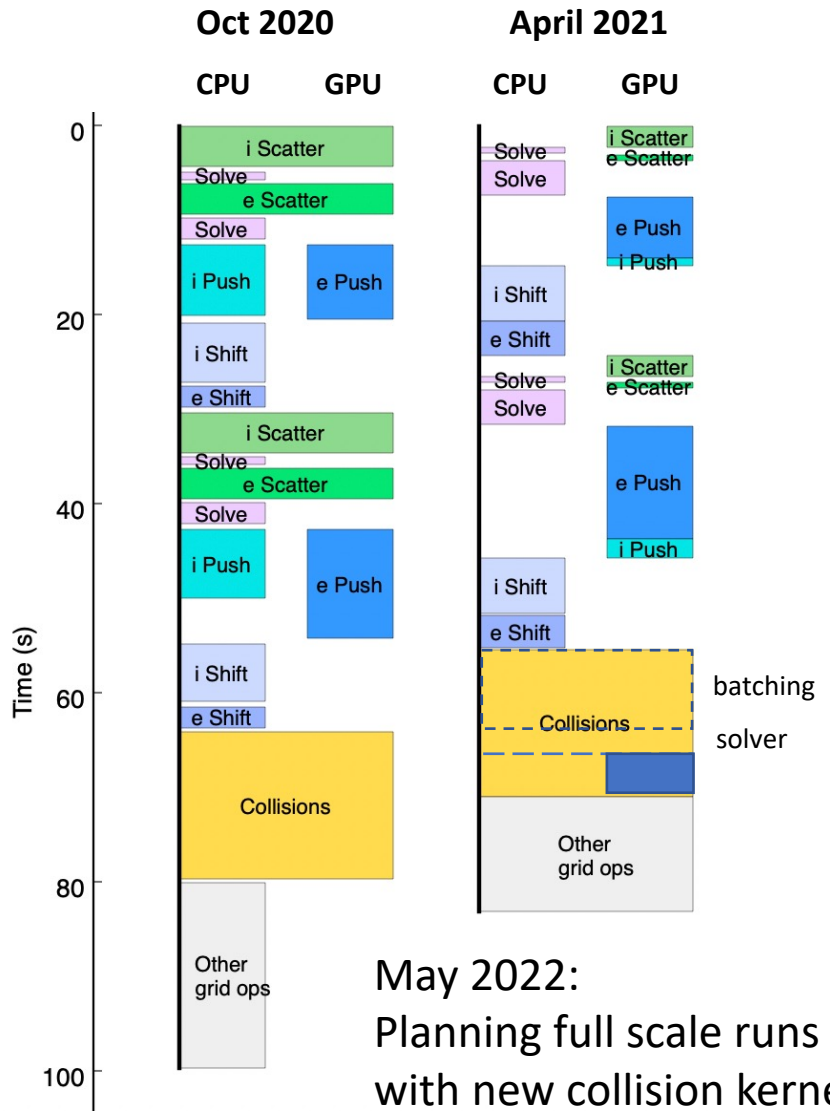
XGC Collision Kernel and ECP Ginkgo

Speed up



XGC Perf. Optimization – Conclusion





May 2022:
Planning full scale runs currently with new collision kernel