OPAL an Open Source Charged Particle Accelerator Simulation Framework

A. Adelmann for the OPAL developer team

ANL - 3. November 2017
The OPAL Developer Team

Ch. Metzger-Kraus  S. L. Sheehy  V. Rizzoglio (*)  D. Winklehner  U. Locans  P. Xiaoying
J. Snuverink  A. Gsell  M. Frey (*)  S. Russell  Y. Ineichen
Ch. Rogers  Ch. Wang  A. Adelmann
OPAL is open source ...
Outline

1. Overview on OPAL
   - History
   - OPAL in a Nutshell
   - Dynamic Kernel Scheduler (DKS) & Examples
   - Multi-Objective Optimisation
   - Iterative Poisson Solver SAAMG-PCG
   - Benchmarking

2. OPAL in Action

3. OPAL V.2.0 & Beyond
History

- started out with my PhD on **3D simulations of space charge effects in Cyclotrons**
- re-branding from MAD9p to OPAL
- open-source
- jumping on to the wagon
  - LANL
  - CIAE
  - RAL
  - MIT
- OPAL 1.6.1 released October 2017 (last 1.x)
- OPAL 2.x (to be released)
  - GA-optimiser integrated
  - full 3D element placement possible
  - partial GPU support
  - wiki manual
History

- started out with my PhD on 3D simulations of space charge effects in Cyclotrons
- re-branding from MAD9p to OPAL
- open-source
- jumping on to the wagon
  - LANL
  - CIAE
  - RAL
  - MIT
- OPAL 1.6.1 released October 2017 (last 1.x)
- **OPAL 2.x** (to be released)
  - GA-optimiser integrated
  - full 3D element placement possible
  - partial GPU support
  - wiki manual
OPAL V.2.0 in a Nutshell

OPAL is an open-source tool for charged-particle optics in large accelerator structures and beam lines including 3D space charge, particle matter interaction, partial GPU support and multi-objective optimisation.

- OPAL is built from the ground up as a parallel application exemplifying the fact that HPC (High Performance Computing) is the third leg of science, complementing theory and the experiment.
- OPAL runs on your laptop as well as on the largest HPC clusters.
- OPAL uses the MAD language with extensions.
- OPAL is written in C++, uses design patterns, easy to extend.
- Webpage: https://gitlab.psi.ch/OPAL/src/wikis/home
- the OPAL Discussion Forum: https://lists.web.psi.ch/mailman/listinfo/opal
- O(40) users
2 OPAL flavours, OPAL-T & OPAL-CYCL released

- Common features
  - 3D space charge
  - particle Matter Interaction (protons)
  - multi-objective optimisation
  - from e, p to Uranium (q/m is a parameter)

1 OPAL-T
  - OPAL-T with time as the independent variable, can be used to model beamlines, rf-guns, injectors
  - many more linac features like auto-phasing, wake fields, 1D CSR

2 OPAL-CYCL (+ FFAG’s)
  - neighbouring turns
  - time integration, 4th-order RK, LF, adaptive schemes
  - find matched distributions with linear space charge
  - spiral inflector modelling with space charge

[M. Toggweiler, AA, et al. (2014)]
2 OPAL flavours, OPAL-T & OPAL-CYCL released

- **Common features**
  - 3D space charge
  - particle Matter Interaction (protons)
  - multi-objective optimisation
  - from e, p to Uranium (q/m is a parameter)

1. **OPAL-T**
   - OPAL-T with time as the independent variable, can be used to model beamlines, rf-guns, injectors
   - many more linac features like auto-phasing, wake fields, 1D CSR

2. **OPAL-CYCL (+ FFAG’s)**
   - neighbouring turns
   - time integration, 4th-order RK, LF, adaptive schemes
   - [M. Toggweiler, AA, et al. (2014)]
   - find matched distributions with linear space charge
   - spiral inflector modelling with space charge
2 OPAL flavours, OPAL-T & OPAL-CYCL released

- Common features
  - 3D space charge
  - particle Matter Interaction (protons)
  - multi-objective optimisation
  - from e, p to Uranium (q/m is a parameter)

1 OPAL-T

- OPAL-T with time as the independent variable, can be used to model beamlines, rf-guns, injectors
- many more linac features like auto-phasing, wake fields, 1D CSR

2 OPAL-CYCL (+ FFAG’s)

- neighbouring turns
- time integration, 4th-order RK, LF, adaptive schemes
  [M. Toggweiler, AA, et al. (2014)]
- find matched distributions with linear space charge
- spiral inflector modelling with space charge
Vlasov-Poisson Equation

Addressing the multi scale challenge

When neglecting collisions, and taking advantage of the electrostatic approximation, the Vlasov-Poisson equation describes the (time) evolution of the phase space $f(x, v; t) > 0$ when considering electromagnetic interaction with charged particles.

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + v \cdot \nabla_x f + \frac{q}{m} \left( E(x, t) + v \times B(x, t) \right) \cdot \nabla_v f = 0. \quad (1)$$

**Solving with ES-PIC**

- Hockney and Eastwood, $h_x(t), h_y(t), h_z(t)$, $M = M_x \times M_y \times M_z$
- SAAMG-PCG solver with geometry (later)
- change $M$ during simulation (many different field solver instances)
- change $\Delta t$ adaptively [M. Toggweiler, AA, et al. (2014)]
- modern computational architectures (later)
Software Architecture

MPI based + HW accelerators + Optimiser

OPAL
- MAD-Parser
- Flavors: t, Cycl
- Distributions
- Solvers: Direct, MG
- Integrators
- PMI, WFC
- FFT
- D-Operators
- NGP, CIC, TSI
- Fields
- Mesh
- Particles
- Load Balancing
- Domain Decomp.
- Communication
- Particle-Cache
- PETE
- Trillions Interface

DKS
- MC
- CUDA
- cuFFT
- cuBLAS

Pilot
- Multi-Objective Optimizer
  - Genetic Optimization Algorithms (NSGA-II)

Trilinos & GSL

OPAL an Open Source Charged Particle Accelerator Simulation Framework
DKS in a Nutshell I

Dynamic Kernel Scheduler (DKS) is a slim software layer between host application and hardware accelerator

- Ease the use of hardware accelerators (GPUs and Intel MICs)
- Fully OO (C++) using CUDA, OpenCL and OpenMP to handle device specific code
- The host application remains portable and adapt better to new hardware that comes available
- Separating the hardware specific code from host application
- Software investment protection - no device code in host application
DKS in a Nutshell II

DKS concept

- **Communication**: common interface to communicate with different types of devices hiding all the details of different frameworks used for each device
- **Function library**: library of predefined algorithms written using CUDA, OpenCL, OpenMP
- **Auto-tuning**: based on the system setup and executable tasks select appropriate implementation and configuration to execute the code (not yet available)
Example 1: A Direct FFT-Based Poisson Solver

Assume you know $G$ the Green’s function

The solution of the Poisson’s equation

$$\nabla^2 \phi = -\rho/\varepsilon_0,$$

for the scalar potential, $\phi$ can be expressed as:

$$\phi(x, y, z) = \int \int \int dx' dy' dz' \rho(x', y', z') G(x - x', y - y', z - z'), \quad (2)$$

where $G$ is the Green function and $\rho$ is the charge density.

Discretisation of Eq. (2) on a grid with cell sizes $h_x, h_y$ and $h_z$ leads to:

$$\phi_{i,j,k} = h_x h_y h_z \sum_{i'=1}^{M_x} \sum_{j'=1}^{M_y} \sum_{k'=1}^{M_z} \rho_{i',j',k'} G_{i-i',j-j',k-k'}, \quad (3)$$

The solution of Eq. (3) can be obtained using FFT based convolution:

$$\phi_{i,j,k} = h_x h_y h_z \text{FFT}^{-1}\{(\text{FFT}\{\rho_{i,j,k}\}) \otimes (\text{FFT}\{G_{i,j,k}\})\}$$
### FFT Poisson solver - results

**Example:** simulation for the PSI Ring Cyclotron.  
**Host code 8 cores:** 2x Intel Xeon Processor E5-2609 v2  
**Accelerator:** Nvidia Tesla K20 or Nvidia Tesla K40

<table>
<thead>
<tr>
<th>FFT size</th>
<th>DKS</th>
<th>Total time (s)</th>
<th>OPAL speedup</th>
<th>OPAL</th>
<th>FFTPoisson time (s)</th>
<th>FFFTPoisson speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>64x64x32</td>
<td>no</td>
<td>324.98</td>
<td></td>
<td>22.53</td>
<td>7.42</td>
<td>×3</td>
</tr>
<tr>
<td></td>
<td>K20</td>
<td>311.17</td>
<td>×1.04</td>
<td></td>
<td>7.42</td>
<td>×3</td>
</tr>
<tr>
<td></td>
<td>K40</td>
<td>293.7</td>
<td>×1.10</td>
<td></td>
<td>7.32</td>
<td>×3</td>
</tr>
<tr>
<td>128x128x64</td>
<td>no</td>
<td>434.22</td>
<td></td>
<td></td>
<td>206.73</td>
<td></td>
</tr>
<tr>
<td></td>
<td>K20</td>
<td>262.74</td>
<td>×1.6</td>
<td></td>
<td>32.15</td>
<td>×6.5</td>
</tr>
<tr>
<td></td>
<td>K40</td>
<td>245.08</td>
<td>×1.8</td>
<td></td>
<td>25.87</td>
<td>×8</td>
</tr>
<tr>
<td>256x256x128</td>
<td>no</td>
<td>2308.05</td>
<td></td>
<td></td>
<td>1879.84</td>
<td></td>
</tr>
<tr>
<td></td>
<td>K20</td>
<td>625.37</td>
<td>×3.6</td>
<td></td>
<td>202.63</td>
<td>×9.3</td>
</tr>
<tr>
<td></td>
<td>K40</td>
<td>542.73</td>
<td>×4.2</td>
<td></td>
<td>160.87</td>
<td>×11.7</td>
</tr>
<tr>
<td>512x512x256</td>
<td>no</td>
<td>3760.46</td>
<td></td>
<td></td>
<td>3327.14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>K40</td>
<td>716.86</td>
<td>×5.2</td>
<td></td>
<td>302.49</td>
<td>×11</td>
</tr>
</tbody>
</table>
Example 2: Degrader for proton therapy

PROSCAN facility
Beam line toward Gantry-3

COMET-cyclotron
(proton - 250 MeV)

Graphite degrader
(230 - 70 MeV)
Example 2: Degrader for proton therapy

- The PSI COMET cyclotron deliver a proton beam at a fixed energy of 250 MeV. For proton therapy it is necessary to decrease the particle energy with in the range of 70 - 250 MeV.
- A degrader is a slab of matter with a thickness adjusted to the amount of energy to be lost.
- Scattering: including Multiple Coulomb Scattering and large angle Rutherford Scattering.
MC simulations for the degrader - results

**Example:** OPAL 1cm thick graphite degrader example.

**Host code:** 2x Intel Xeon Processor E5-2609 v2

**Accelerator:** Nvidia Tesla K20, K40 or Intel Xeon Phi 5110p

<table>
<thead>
<tr>
<th>Particles</th>
<th>DKS</th>
<th>Degrader time (s)</th>
<th>Degrader speedup</th>
<th>Integration time (s)</th>
<th>Integration speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^5$</td>
<td>no</td>
<td>20.30</td>
<td></td>
<td>3.46</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MIC</td>
<td>2.29</td>
<td>×8</td>
<td>0.89</td>
<td>×4</td>
</tr>
<tr>
<td></td>
<td>K20</td>
<td>0.28</td>
<td>×72</td>
<td>0.15</td>
<td>×23</td>
</tr>
<tr>
<td></td>
<td>K40</td>
<td>0.19</td>
<td>×107</td>
<td>0.14</td>
<td>×24</td>
</tr>
<tr>
<td>$10^6$</td>
<td>no</td>
<td>206.77</td>
<td></td>
<td>34.93</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MIC</td>
<td>5.38</td>
<td>×38</td>
<td>4.62</td>
<td>×7.5</td>
</tr>
<tr>
<td></td>
<td>K20</td>
<td>1.41</td>
<td>×146</td>
<td>1.83</td>
<td>×19</td>
</tr>
<tr>
<td></td>
<td>K40</td>
<td>1.18</td>
<td>×175</td>
<td>1.21</td>
<td>×29</td>
</tr>
<tr>
<td>$10^7$</td>
<td>no</td>
<td>2048.25</td>
<td></td>
<td>351.64</td>
<td></td>
</tr>
<tr>
<td></td>
<td>K20</td>
<td>14.4</td>
<td>×142</td>
<td>17.21</td>
<td>×20</td>
</tr>
<tr>
<td></td>
<td>K40</td>
<td>12.79</td>
<td>×160</td>
<td>11.43</td>
<td>×30</td>
</tr>
</tbody>
</table>
Multi-Objective Optimisation with OPAL


- Access to all OPAL statistics data as objectives
- Access to all OPAL variables as design variables
- Specify the MOOP in the OPAL input file
- Finds Pareto optimal solutions (NSGA-II)
- No tight coupling to parallelisation mechanism
- No tight coupling to optimisation algorithm
- Runs smoothly with 10000 cores and hopefully more
Iterative Poisson Solver SAAMG-PCG

Boundary Problem

\[ \Delta \phi = -\frac{\rho}{\varepsilon_0}, \text{ in } \Omega \subset \mathbb{R}^3, \]
\[ \phi = 0, \text{ on } \Gamma_1 \]
\[ \frac{\partial \phi}{\partial n} + \frac{1}{d} \phi = 0, \text{ on } \Gamma_2 \]

- \( \Omega \subset \mathbb{R}^3 \): simply connected computational domain
- \( \varepsilon_0 \): the dielectric constant
- \( \Gamma = \Gamma_1 \cup \Gamma_2 \): boundary of \( \Omega \)
- \( d \): distance of bunch centroid to the boundary

\( \Gamma_1 \) is the surface of an
- elliptic beam-pipe
- arbitrary beam-pipe element
Iterative Poisson Solver \textbf{SAAMG-PCG} cont.

We apply a second order finite difference scheme which leads to a set of linear equations

$$Ax = b,$$

where $b$ denotes the charge densities on the mesh.

- solve anisotropic electrostatic Poisson PDE with an iterative solver
- accuracy $\varepsilon$ is a parameter
- reuse information available from previous time steps
- achieving good parallel efficiency
- irregular domain with “exact” boundary conditions
- easy to specify boundary surface
Benchmarking
AWA-Gun Code Comparison - N. Neveu (ANL & IIT)

All codes matched within 5%. Well below measurement thresholds at AWA.
Outline

1. Overview on OPAL

2. OPAL in Action
   - A Selection of Past Achievements
   - S2E ERL bERLinPro
   - New FFAG modelling capabilities in OPAL

3. OPAL V.2.0 & Beyond
A Selection of Past Achievements

Precise high intensity cyclotron modelling
[Y. Bi, AA, et al., PR-STAB 14(5) (2011)]

Neighbouring bunch modelling
[J. Yang, AA, et al., PR-STAB 13(6) (2010)]

Realistic Injection Simulations of a Cyclotron Spiral Inflector

BerlinPRO - S2E Model
[B. Kuske, M. Abo-Bakr (2017)]
500-kV Low-Emittance Electron Source

[T. Schietinger et.al.]
The SwissFEL Injector Test Facility Gun

[T. Schietinger et.al.]
S2E ERL bERLinPro
Triggered a full 3D version of OPAL-T (lead by Ch. Metzger)

Solution: place elements in 3D space.

- include all elements only once although traversed twice ($n \times$)
- add apertures to all elements
- add origin and initial orientation of beamline (elements)
New OPAL Element **Ring**
Mostly contributed by C. Rogers

OPAL requires modification to adequately track FFAG field maps

- OPAL-t allows tracking through a set of beam elements in linac-type geometry
- OPAL-cycl previously hard coded to use 2D mid plane field map + single RF cavity
- Aim to introduce the capability to track through a set of *arbitrary* beam elements in ring-type geometry
- Additionally introduce specific capability to track through a 3D field map in a sector-type geometry

Use ERIT ring as test-bed for this development
Getting closed orbit through OPAL

All ERIT simulations are curtesy of C. Rogers
Dynamic Apperture

\[ \Delta t = 0.1 \text{ ns} \]

- After 100 turns aperture looks as expected in OPAL
- Is this dynamic aperture or field map aperture?
- Field map extent is ± 250 mm in x
- Particles outside aperture are lost after < 1 turn
Benchmarking

[S. L. Sheehy, et al]
Outline

1. Overview on OPAL
2. OPAL in Action
3. OPAL V.2.0 & Beyond
OPAL V.2.0 & Beyond

- Consolidation has begun with V 2.0
  - major rewrite of OPAL-T, Distribution class
  - major rewrite of OPAL-CYCL, (ongoing)
  - strong typing, versioning of input files
  - no multipacting
  - no envelope model
  - Manual is converted to Wiki
  - gitlab and issue tracker

- New features
  - curved multipoles (FFAG and Proton Therapy Modelling)
  - maps tracking ($\mathcal{M} +$ space charge)
  - $\mathcal{M}_{sc}$ based on moments of the distribution
  - OPAL Simulation in the Cloud
  - AMR-Fieldsolver (BoxLibX)
  - Collisions
OPAL Simulation in the Cloud
D. Bruweiler (RadiaSoft)

- No need to download and maintain codes
- Codes are preinstalled and free to use
- 7 particle accelerator codes
  - Beams: Elegant, OPAL, Synergia, Warp
  - X-Rays: SRW, Shadow3
  - FEL: Genesis v.2
- Made possible by Docker technology
  - portable archive of code & dependencies
- Linux - codes run at native speeds
- Mac OS X and Windows:
  - Docker provides a lightweight VM

Sirepo & Jupyter development supported by U.S. DOE
OPAL Simulation in the Cloud
D. Bruweiler (RadiaSoft)

Sirepo: Browser-based GUIs for codes

- http://sirepo.com
- 5 particle accelerator codes
  - Elegant, Warp, SRW, Shadow3, Hellweg
  - OPAL coming soon!
- Share simulations
  - via links, python code, or Jupyter Notebooks
- Nothing to compile or install
- Fast, interactive scientific plotting
AMR and OPAL-cycl

- **BoxLib** based AMR
- PhD. project SNF funded
- Focus on PSI-Ring neighboring bunch interaction & UQ
  [arXiv:1509.08130]
Collisions I

Motivation

1. model emission of ultra cold electrons
2. understand Coulomb scattering (Borsch effect) [J. Qiang, et.al]
3. do we have to worry about it in next generation machines?
4. model non Gaussian tails (high intensity hadron machines)

In [J. Qiang, et.al] we wrote:

- Nano-tips with high acceleration gradient around the emission surface have been proposed to generate high brightness beams.
- However, due to the small size of the tip $r = 10 \text{ nm}$, the charge density near the tip is very high even for a small number of electrons.
- The stochastic Coulomb scattering near the tip can degrade the beam quality and cause extra emittance growth and energy spread.
Collisions II
Motivation

Using a brute force $N^2$ summation we obtained the following observations:

- slice emittance over the bunch length $\times 2$ higher
- energy spread $\times 100$ higher
Collisions III

Motivation

The

$$P^3M = \text{Particle-Particle} + \text{Particle-Mesh}$$

is an efficient way to accomplish this task.

- high resolution from PP part: $O(K^2), K \ll N, 1/(|x - x'| + \varepsilon)$
- good performance from PM part: $\Phi(x) = \int G(x, x')\rho(x')d^3x'$
- adjustable influence of Coulomb collisions by fixing $K$ in choosing $r_c$

Opens up the possibility of S2E beam simulations with *adjustable* Coulomb interaction
Disorder Induced Heating

Problem Setup

Loosely connected to LBL UED parameters:

- spherical, cold beam of radius $R = 17.7400 \, \mu m$ and charge $Q = 25 \, fC$
- constant focusing applied
- cubical domain with edge length $L = 100 \, \mu m$
- $P^3M$ simulation over 5 plasma periods
- boundary conditions: open in $x, y$ periodic in $z$
- $M = 256^3$
- $r_c$ varying from $0 \, \mu m$ to $3.1250 \, \mu m$
- $N = \mathcal{O}(10^7)$
- $P_{core} = 128$
- simulation over 1000 time-steps

Goal compute $T^f$
Disorder Induced Heating

Simulation Results MSc. thesis B. Ulmer

![Graph showing normalized x-emittance over number of plasma periods for different values of \( r_c \).]
Disorder Induced Heating
Simulation Results MSc. thesis B. Ulmer

- coulomb collisions are responsible for beam heating!
- P3M can reproduce analytic results
References I


[AA et al., CPC (2016)] A. Adelmann, et al., CPC, dx.doi.org/10.1016/j.cpc.2016.05.013


[B. Kuske, Ch. Metzger-Kraus] IPAC2016, Busan, South Korea, accelconf.web.cern.ch/AccelConf/ipac2016/papers/thoba03.pdf

[Ch. Metzger-Kraus et.al] IPAC2016, Busan, South Korea, accelconf.web.cern.ch/AccelConf/ipac2016/papers/wepoy034.pdf


[J. Qiang, et.al] IPAC 2007, 10.1109/PAC.2007.4441024