

Modern Computational Accelerator Physics

James Amundson Alexandru Macridin *Panagiotis Spentzouris*

Fermilab

USPAS January 2015

Synergia

- Accelerator simulation package
 - independent-particle physics
 - linear or nonlinear
 - collective effects
 - simple or computationally intensive
 - can go from simple to complex, changing one thing at a time
- Goal: best available physics models
 - *best* may or may not mean *computationally intensive*

<https://cdcvns.fnal.gov/redmine/projects/synergia2/wiki>

- Designed for range of computing resources
 - laptops and desktops
 - clusters
 - supercomputers
- Goal: best available computer science for performance
 - significant interaction with computer science community

Synergia is developed and maintained by the
Scientific Computing Division's
Accelerator Simulation group

James Amundson, Paul Lebrun, Qiming Lu, Alex Macridin, Leo Michelotti
(CHEF), Chong Shik Park, Panagiotis Spentzouris and Eric Stern

With development contributions from Tech-X: **Steve Goldhaber**

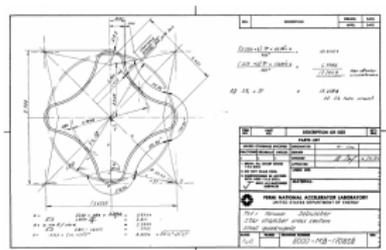
- Single-particle physics are provided by CHEF
 - direct symplectic tracking
 - magnets, cavities, drifts, etc.
 - (and/or) arbitrary-order polynomial maps
 - many advanced analysis features
 - nonlinear map analysis, including normal forms
 - lattice functions (multiple definitions)
 - tune and chromaticity calculation and adjustment
 - etc.
- Apertures
- Collective effects (single and multiple bunches)
 - space charge (3D, 2.5D, semi-analytic, multiple boundary conditions)
 - wake fields
 - can accommodate arbitrary wake functions

Variety of boundary conditions and levels of approximation

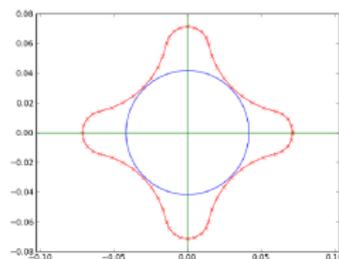
- 3D open transverse boundary conditions
 - Hockney algorithm
 - open or periodic longitudinally
- 3D conducting rectangular transverse boundary
 - periodic longitudinally
- 3D conducting circular transverse boundary
 - periodic longitudinally
- 2.5D open boundary conditions
 - 2D calculation, scaled by density in longitudinal slices
- 2D semi-analytic
 - uses Bassetti-Erskine formula
 - σ_x and σ_y calculated on-the-fly
- New space charge models can be implemented by the end user

Synergia aperture model

- Apertures can be associated with elements and/or steps
- 2D model
 - could be extended with slices
- Geometric
 - circular
 - elliptical
 - rectangular
 - polygon
 - wire
- Abstract
 - phase space
 - Lambertson
 - removes particles
- New apertures can be implemented by the end user



Engineering drawing of FNAL
Debuncher quad cross section



Synergia implementation:
detailed, but fast
(inscribed circle optimization)

Synergia 2.1 design

- Synergia 2.1 is a major milestone
 - very different from Synergia 1
 - significantly different from Synergia 2
 - designed for widespread use
- Synergia is a mix of C++ and Python
 - all computationally-intensive code is written in C++
 - user-created simulations are usually written in Python
 - pure-C++ simulations are possible
- Synergia provides a set of functions and classes for creating simulations
 - many examples available
- **Virtually every aspect of Synergia is designed to be extendable by the end-user**
 - code in C++ and/or Python

Synergia simulations

- A simulation consists of propagating a *Bunch* (or *Bunches*) through a *Lattice*.
- Inputs: machine lattice, initial bunch parameters



- *Diagnostics*

- 6D means
- 6D std deviations
- 6x6 covariance matrix
- 6x6 correlation matrix
- individual particle tracks
- dump of all particles
- losses at locations in lattice
- can be extended. . .

- *Actions* can specify when *Diagnostics* will be applied

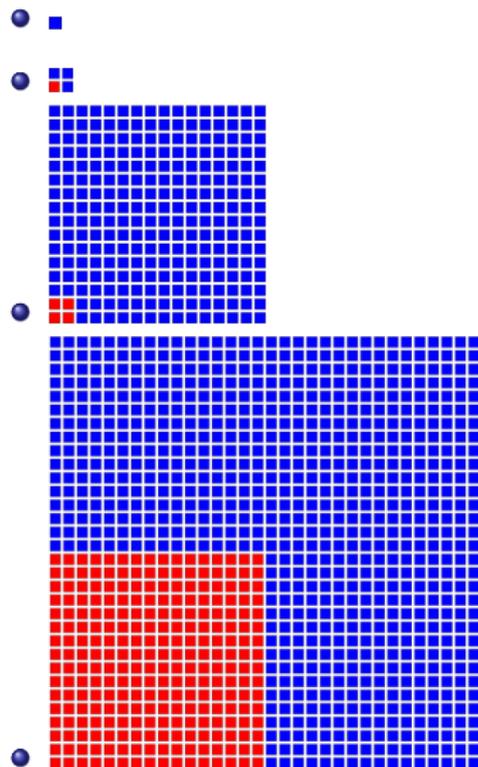
- every n steps
- every m turns
- at specified sets of steps
- at specified sets of turns
- by user-specified logic
- more

Feature: checkpointing

- Synergia simulations can be saved to disk (checkpointed) at any point
 - allows recovery from hardware failure
 - allows jobs that take longer than batch queue limits
- All simulation objects can be checkpointed
 - even, e.g., objects with open files
- Checkpointing available for both C++ and Python objects
 - *including end-user objects*
- User specifies parameters
 - every n turns
 - do p out of q total turns
 - send a message to stop at the end of next turn

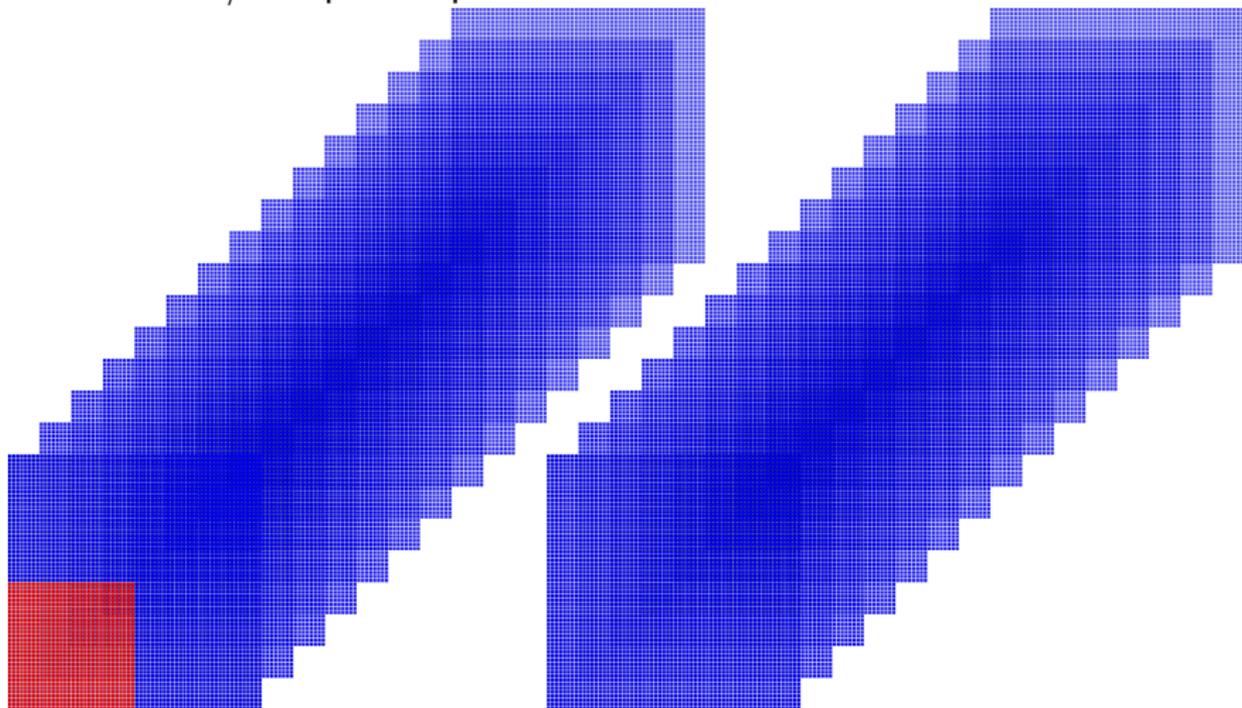
Feature: scalability

- Why?
 - statistics (many particles)
 - multiple bunches
 - take advantage of modern computing resources
- I use Synergia every day on a single CPU
- Synergia can take advantage of multiple cores on a single CPU
- We regularly run Synergia using 256 cores of a Linux cluster
- Single-bunch Synergia simulations scale well to over 1024 cores on supercomputers such as IBM BlueGene or Cray XT



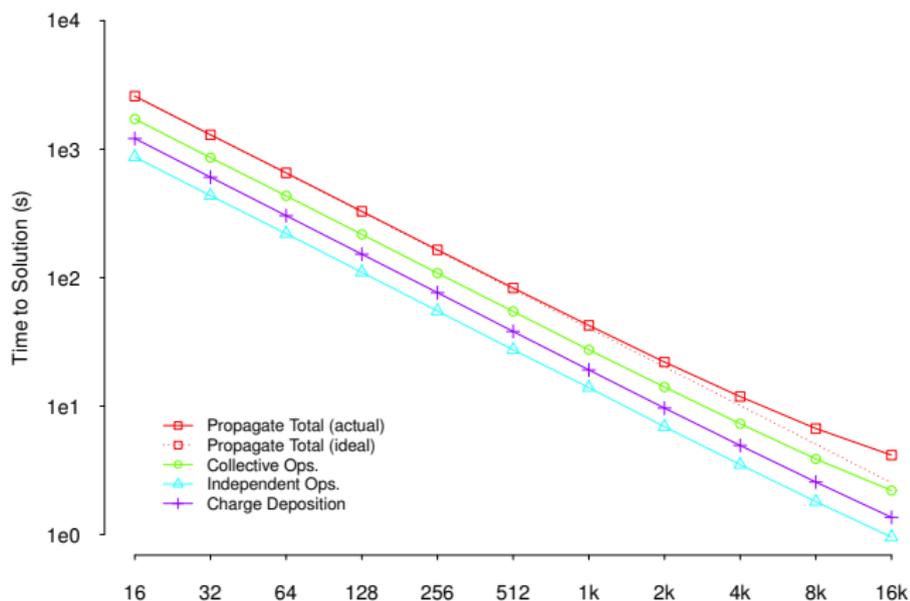
Weak scaling

- Multi-bunch Synergia simulations have been shown to scale to 131,072 cores on Argonne Leadership Computing Facility's Intrepid, a BlueGene/P supercomputer

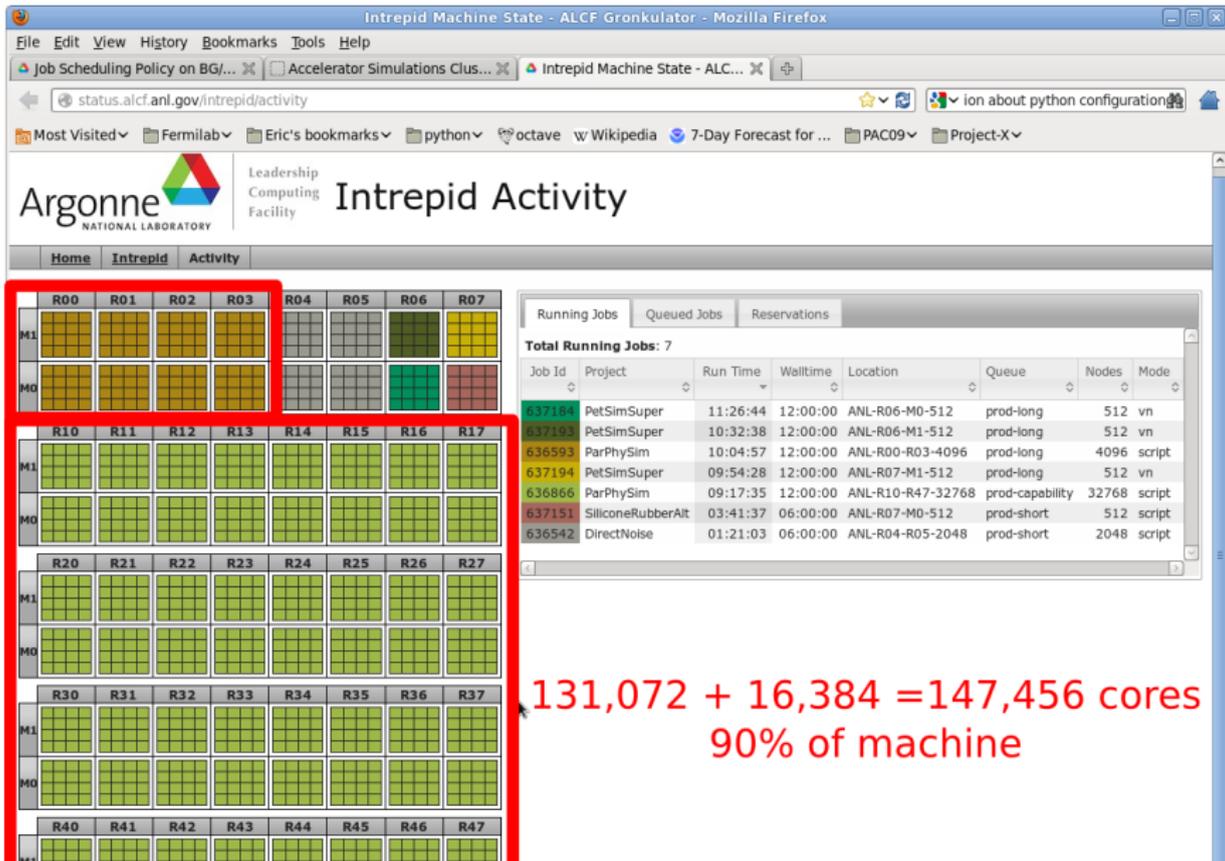


Strong scaling

- Single-bunch Synergia simulations have been shown to scale past 8192 cores on ALCF's Mira, a BlueGene/Q supercomputer
 - Strong scaling, *i.e.*, fixed problem size
 - $(32 \times 32 \times 1024)$ grid, 100 grid cells per particle, trivial apertures)



Main Injector + Booster = 90%



131,072 + 16,384 = 147,456 cores
90% of machine