1 Heavy Ion Beams and Interactions with Plasmas and Targets (HEDLP and IFE)

Principal Investigator: Alex Friedman, LLNL and LBNL
Authors: A. Friedman, R. H. Cohen, D. P. Grote, W. M. Sharp, *LLNL and LBNL*;
I. D. Kaganovich, *PPPL*; A. E. Koniges and W. Liu, *LBNL/NERSC*
NERSC Repositories: MP42 (HIFS-VNL) and M74 (PPPL Non-neutral Plasmas)

1.1 Project Description

1.1.1 Overview and Context

DOE's Fusion Energy Sciences program supports research on the science of intense heavy-ion beams, on high-energy-density physics (HEDP) and especially Warm Dense Matter (WDM) generated by ion beams, and on target physics for ion-beam-driven Inertial Fusion Energy. The major participants in this endeavor are the partners in the Heavy Ion Fusion Science Virtual National Laboratory (HIFS-VNL), a collaboration of LBNL, LLNL, and PPPL, and colleagues at universities. Ongoing experiments are focused on generating, compressing, and focusing intense, space-charge-dominated ion beams and using them to heat thin foils, the evolution and properties of which are then measured. To further this work, a new accelerator facility, the Neutralized Drift Compression Experiment-II (NDCX-II), has been built at LBNL, and is currently undergoing commissioning. Obtaining maximum benefit from these experiments is a key near-term goal of the simulation program.

Simulation efforts in support of NDCX-II have concentrated on developing the physics and engineering design of the accelerator, on identifying favorable operating points, and on planning the Warm Dense Matter experiments to be done with its beam once full operations ensue. To support machine operations and user experiments, the scope of our simulations is evolving, with primary emphasis on detailed simulations of the actual beams as realized and of their interactions with various targets. This includes extensive studies of the coupled dynamics of the beam and a neutralizing plasma (which allows the beam to be compressed into a compact volume in space and time). These studies employ the beam dynamics code Warp and other kinetic models, run primarily at NERSC. As NDCX-II shifts emphasis toward studies of the target response, the simulated ion beam data at the target plane will be transferred into hydrodynamic simulations of targets using the Hydra code (run at LLNL) and the ALE-AMR code (run at NERSC).

Intense ion beams are non-neutral plasmas and exhibit collective, nonlinear dynamics that must be understood using the kinetic methods of plasma physics. This physics is rich and subtle: a wide range in spatial and temporal scales is involved, and effects associated with instabilities and non-ideal processes must be understood. In addition, multispecies effects must be modeled during the interaction of the beams with any stray electrons in the accelerator and with the neutralizing plasma in the target chamber. The models must account for the complex set of interactions among the various species, with the walls, and with the applied and self-fields. Finally, oscillations of the beam core and "mismatches" of

the beam confinement can dilute the beam phase space and parametrically pump particles into a low-density outlying "halo" population; this physics imposes stringent requirements on numerical noise, requiring good particle statistics and mesh resolution. A blend of numerical techniques, centered around the Particle-In-Cell method, are used in Warp to address these needs, including: electrostatic and electromagnetic solvers, adaptive mesh refinement, cut-cell boundaries, a large time step particle pusher, and implicit field solvers and particle pushers. Ion beams have a long memory, and initialization of a simulation at mid-system with an idealized particle distribution is often unsatisfactory; thus, a key goal is to further develop and extensively exploit an integrated and detailed source-to-target beam simulation capability.

In order to determine material properties (such as equation of state) in the warm dense matter (WDM) regime, we must use simulation codes to help interpret experimental diagnostics. The WDM regime is at the boundaries of solid-state physics and plasma physics, between non-degenerate and degenerate material, between ionized and neutral states, and between liquid and vapor. To understand and simulate the results of experiments that volumetrically heat material, many processes must be combined into a single simulation. These include phase changes, ionization processes, shock processes, spall processes, and droplet formation, to name a few. One goal is to simulate as effectively as possible the experiments that will be conducted on NDCX-II, as it compresses ion beams and uses them to heat foil targets to temperatures of about 10,000 K. The matter is heated so rapidly that, although its temperature is well above that of vaporization, inertia will keep it at solid density (at least for an inertial confinement time of order one ns). Experimental diagnostics will record the response of the material (e.g. temperature, density, velocity) to infer the equation of state and other properties. Synthetic diagnostics in the hydrodynamic simulations will be essential for inter-comparison.

1.1.2 Scientific Objectives for 2017

Between now and 2017, the project will concentrate on using advanced simulations to provide critical support to the NDCX-II facility, including both its operations and the experiments that it will drive. There is considerable scope for optimization of the beam dynamics in the accelerator, in the neutralized drift line, and in the final transverse focusing system. Already, "ensemble" runs at NERSC have enabled beam optimization, but these have not included first-principles plasma models; by 2017 this should be routine.

NDCX-II was designed to be extensible and reconfigurable; on that timescale, the machine should be extended by the addition of ten additional acceleration cells (which are already available but require modification and installation), significantly increasing the beam kinetic energy and decreasing its pulse duration. This configuration, too, will need extensive simulation, including studies of the neutralization process and of non-ideal effects. Options for multi-pulse operation also need to be thoroughly explored, for "pump-probe" experiments.

A long-standing item on the DOE list of future facilities, the Integrated-Beam High Energy Density Physics Experiment (IB-HEDPX), received CD-0 approval in Dec., 2005. In a synopsis that was recently submitted to the FESAC facilities panel for their consideration (as they provide input to DOE for an updated list), the project has been re-named the IB-HEDPF (with the "F" denoting its role as a user Facility). It is hoped that this NERSC project will include studies of IB-HEDPF at a "physics design" level in 2017.

Hydrodynamic codes such as Hydra and ALE-AMR can be used to model ion deposition and the subsequent response of the target to volumetric heating. In support of HEDLP experiments driven by NDCX-II, a core research effort will conduct hydrodynamic simulations with realistic incoming beams (obtained from both simulations and detailed diagnostics, requiring reconciliation). NDCX-II is a user facility; thus, while some experiments will be carried out by the core group and its close collaborators, more will be fielded by visiting users. Almost all experiments will need at least some local simulation support, even when outside users employ their own codes as their principal simulation tools. LBNL, LLNL and SLAC have recently formed a Bay Area High Energy Density Sciences (BA-HEDS) cooperative to facilitate coordinated experiments, some driven by NDCX-II's ion beam, and others by high-power lasers and X-ray FEL's at the other labs. These developments will enlarge the scope of the simulation program.

The project will also carry out research on elements of target physics for heavy-ion beam-driven IFE, including beam-target energy coupling, new ideas for improved targets, and assessments of target behavior. While much of the actual target design work will be carried out on computer facilities other than NERSC (especially systems at LLNL), it will be important to employ NERSC for other aspects of this program element.

1.2 Computational Strategies (now and in 2017)

1.2.1 Approach

For each of the project goals described above, the strategies to be employed are:
(1) *Optimize the properties of the NDCX-II beam for each class of target experiments; achieve quantitative agreement with measurements; develop improved machine configurations and operating points.* To accomplish these goals, we plan to use Warp to simulate NDCX-II from source to target, in full kinetic detail, including first-principles modeling of beam neutralization by plasma. Additional tools, including kinetic codes such as LSP and BEST, will be employed as appropriate. The output from an ensemble of Warp runs (representing shot-to-shot variations) will be used as input to target simulations using ALE-AMR on NERSC, and other codes on NERSC and elsewhere.
(2) *Develop enhanced configurations of NDCX-II, and carry out studies to enable a next-step ion beam facility (IB-HEDPF).* To accomplish these goals, much of the work will involve iterative optimizations employing Warp. These will, at first, assume ideal beam neutralization downstream of the accelerator, but will then advance to first-principles plasma models.
(3) *Carry out detailed target simulations in the Warm Dense Matter regime using the ALE-AMR code, including surface tension effects, liquid-vapor coexistence, and accurate models of both the driving beam and the target geometry.* For this we will need to make multiple runs (to capture shot-to-shot variations), and to both develop and employ synthetic diagnostics (to enable comparison with experiments). The new science that will be revealed is the physics of the transition from the liquid to vapor state of a volumetrically superheated material, wherein droplets are formed, and wherein phase transitions, surface tension and hydrodynamics all play significant roles in the dynamics. These simulations will enable calculations of equation of state and other material properties, and will also be of interest for their illumination of the science of droplet formation.

1.2.2 Codes and Algorithms

Our main ion-beam code, Warp, was originally developed to simulate space-charge-dominated beam dynamics in induction accelerators for heavy-ion fusion (HIF). In recent years, the physics models in the code have been generalized, so that Warp can model beam injection, complicated boundary conditions, denser plasmas, a wide variety of accelerator "lattice" components, and the non-ideal physics of beams interacting with walls and plasmas. The code now has an international user base and is being applied to projects both within and far removed from the HIF community.

Warp uses a flexible multi-species particle-in-cell model to describe beam dynamics and the electrostatic or electromagnetic fields in particle accelerators. While the core routines of Warp solve finite-difference representations of the field equations and relativistic or non-relativistic motion equations, the code also uses a large collection of subordinate models to describe lattice elements and such physical processes as beam injection, desorption, and ionization. The representation of particles by a much smaller number of "macroparticles" can be derived from Boltzmann's equation, describing the evolution of a population of particles interacting by collisions and the collective fields.

Warp is a 3-D time-dependent multiple-species particle-in-cell (PIC) code, with the addition of a "warped-coordinate" particle advance to treat particles in a curved beam pipe. Self-fields are obtained via Poisson equations for scalar and vector potentials, or via Maxwell equations. Time-dependent applied external fields can be specified through the Python user interface. Warp also has 2-D models, using Cartesian or cylindrical geometry, as well as low-order moment equations. Models are available for background gas, wall effects, stray electrons, space-charge-limited and source-limited emission, and atomic processes such as charge exchange. Elaborate initialization and run-time options allow realistic modeling of complex systems. A beam may be initialized with one of many analytic distributions or with a distribution synthesized from experimental data, or ions can be emitted from a flat or curved diode surface. Lattice-element fields may be represented by several options, from simple hard-edge analytic forms to first-principles 3-D calculations. Poisson's equation can be solved using several methods, including FFT, Multigrid, and AMR/Multigrid. The electromagnetic (EM) solver can also use MR. With multigrid, the Shortley-Weller method for the subgrid-resolution description of conductors allows the use of complicated boundary conditions.

Parallelization of Warp is done using domain decomposition with MPI. Warp uses independent spatial decompositions for particles and field quantities, allowing the particle and field advances to be load-balanced independently. In transverse-slice 2-D runs, the field solution is repeated on each node, but solved in parallel by processors within a node.

The size and duration of Warp jobs varies tremendously, depending on such factors as problem dimensionality, grid size, duration, particle count, and the physical processes being modeled. With our generalized decomposition, we do not foresee any limitation resulting from the code's architecture; but Poisson solution scaling is poor at large problem sizes. For a 3-D test problem using 512 x 512 x 512 cells, we have demonstrated excellent parallel scaling of the electromagnetic PIC capability, up to about 50,000 processors.

Our Warp projects tend not to be data intensive; they use modest amounts of memory but require many time steps. We typically run (in 2-D or 3-D) with of order 100 grid cells along each transverse axis and 1000 grid cells along the longitudinal axis. Large 3-D simulations typically have a mesh of order several 100s by 100s by 1000s of grid cells.

The data per cell is either a several scalars or several 3-D vectors, depending on the field model. Typically of order 10^6 particles are used in runs without plasma electrons, with 13 or more variables per particle, and including dynamics electrons can require up to 30 x 10^6 particles. We currently use 120 to 1920 processors for typical Hopper runs and up to 6144 for a few key runs with fine grids and an augmented number of particles.

ALE-AMR is a relatively new code that combines Arbitrary Lagrangian Eulerian (ALE) hydrodynamics with Adaptive Mesh Refinement (AMR) to connect the continuum to micro-structural regimes. The code is unique in its ability to model both hot radiating plasmas and cold fragmenting solids. The hydrodynamics are done in a Lagrangian model (wherein material moves with the mesh), but the resulting mesh can be modified to prevent tangling or severe mesh distortions. If the entire mesh is restored to the mesh of the previous time-step after every step, the code is said to be run in Eulerian mode (fixed mesh). In general, this is not done, and we only modify a portion of the mesh during a fraction of the time steps. This ability to do selective remapping is the reason to use the word "arbitrary." We also employ the Hydra code, another 3-D radiation hydrodynamics ALE code; that code is run on LLNL computers, which are accessed from LBNL and LLNL by group members. A common feature of ALE codes is the ability to have multiple materials in a given computational zone. Such mixed zones are generally created during the advection phase of the advance, when material from the old mesh is transferred to the new mesh. The ALE-AMR code uses a volume-of-fluids approach to calculate the interface between different materials in a zone. Information from neighboring zones can be used to explicitly construct the interfaces if needed.

One key added capability of ALE-AMR, relative to other ALE codes such as Hydra, is the ability to dynamically add mesh elements (refinement) or remove mesh elements (coarsening) during the run. This capability is called Adaptive Mesh Refinement (AMR); however, the ability to remove zones by coarsening the mesh when there are no longer any steep gradients is also important. ALE-AMR refines by a factor of three along each dimension, so in 3D one zone becomes 27 zones. During refinement all material interfaces must be explicitly defined to place the correct amount of each material in the new zones. Numerical techniques were developed for many of the physics packages to work efficiency on a dynamically moving and adapting mesh. ALE-AMR also continues several features that allow for very long-time simulations, a unique fragmentation capability, and the ability to "shape-in" unusual objects.

Additional physics, beyond basic hydrodynamics, is implemented in ALE-AMR using operator splitting. For example, a flexible strength/failure framework allows "pluggable" material models to update the anisotropic stress tensor that is used in the hydro advance during the following step. The code also includes an ion-deposition model for bulk heating of the material, and both heat conduction and radiation transport using the diffusion approximation. The hydro uses explicit time stepping but some packages, e.g., radiation transport, can do an implicit solve at each explicit time step.

The parallelism in ALE-AMR is currently MPI-only with the ability to do dynamic load balancing based on the computational requirements. The domain decomposition is zonal. During the ion deposition phase of the simulation, the regions with ion beams will have smaller number of zones in the domain assigned to a given processor because of the additional computation work associated with beam energy deposition. There are various

places in the code were additional levels of parallelism are possible and we are investigating hybrid models, e.g., OpenMP + MPI.

1.3 HPC Resources Used Today

1.3.1 Computational Hours

NERSC is (far and away) our principal computational resource. Computer resources at LLNL are used for HYDRA. To date these have been modest, but it is anticipated that this usage will expand as NDCX-II goes into production mode. Computational clusters at LBNL and PPPL also are employed for code development and testing, and for some of the same purposes as NERSC when the computational demands are much smaller.

1.3.2 Data and I/O

Scratch (temporary) space: few 100 GB
Permanent (can be shared, NERSC Global Filesystem /project): 100 GB (including Home)
HPSS permanent archival storage: 1 TB

As noted above, our project does not tend to be storage intensive. During computations, data is primarily written to scratch, which we also use for short-term storage (few weeks) as runs are analyzed. For speed in the loading of Python (a topic of current interest), we keep Python installations in scratch (with different ones for dynamic and static loading, for example). We've been keeping our code repositories (for Warp and supporting tools) in /project. In Home, we keep smaller output data files for long-term reference and post-processing. HPSS is used for storing full output datasets, restart dumps, and large graphics files and data dumps.

I/O is of two general types: gathering data into the first processor and then writing it to disk; and direct writes by each processor into its own file.

With regard to constraints due to I/O, we have found that loading Python onto a large number of processors scales poorly. NERSC staff has helped us improve performance here, but it remains an area of concern for future machines in the 2017 time frame.

1.3.3 Parallelism

The number of compute cores is highly problem dependent – Warp in particular is used for many different kids of runs. While we have used tens of thousands of cores, in a typical large run we use a few hundred to a few thousand. The largest number we would consider using in a production run, today, is 100,000. Fewer are typically used because the problems being solved do not require more. Also we need to conserve our allocation and minimize human effort, so for example, "ensemble" runs (many independent simulations in a single batch submission) use relatively small numbers of particles, at the price of jitter in the results. We sometimes have multiple jobs (ensemble or otherwise) running concurrently (up to about five). With regard to strong vs. weak scaling, this depends upon the field solver being used, e.g., the Poisson solver does not scale well to large numbers of cores, while the electromagnetic solver does. We want strong scaling for increased resolution, rather than for an enlarged problem domain. We are currently investigating the use of communication-reduced solvers in the context of Warp.

1.4 HPC Requirements in 2017

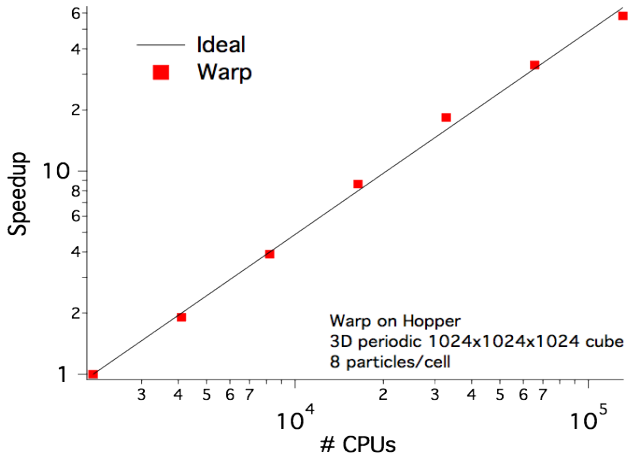
1.4.1 Computational Hours Needed

We anticipate a need for (of order) 40 MCRU's / year in the 2017 time frame, primarily driven by the need for computationally larger runs involving detailed plasma descriptions, by the need to run ensembles for optimization and sensitivity analysis, and by the desire to be able to use improved methods for optimization (large particle counts are needed in order to have a smooth optimization cost function). We also anticipate much greater use of the ALE-AMR code to support a wide variety of experiments on NDCX-II.

Our need for NERSC time is rapidly growing. In past years our requirements were modest. We concentrated on NDCX-II machine design using simplified tools such as a 1-D beam physics code (ASP), and on Warm Dense Matter simulations using both Hydra (at LLNL) and a specialized 1-D code (Dish). In FY11, we began applying NERSC resources to iterative design calculations for the NDCX-II facility, and our usage rate increased roughly five-fold. Now, three developments compel us to carry out far more demanding simulations at NERSC: (1) the need to capture beam-in-plasma effects (requiring far more simulation particles, and a smaller time-step size and grid spacing); (2) Our interest in self-focusing of ion beams in plasma will require use of our explicit electromagnetic solver, which requires a substantially smaller time step due to the Courant condition; and (3) the introduction of ttarget-response studies (requiring considerable resources for realistic problems).

We begin with a discussion of recent Warp usage, which has emphasized iterative design and assessment (on NERSC and LBNL clusters) using ensembles of runs with random errors. For this task 256 cases (instances) are typically run in a single batch job. The number of cores has ranged between 768 and 6144, depending on the problem, with less than 1 GB/core, using 60 GB total memory and 4 hours of wall-clock time. Much data processing is in-line, and I/O is only about 100 GB / batch job. This approach leads to very light traffic in and out of NERSC, with results stored at the Center.

Another class of Warp runs models ion beams in plasmas. Current problems of this type are axisymmetric runs using 100's x 1000's of cells and tens of millions of particles (with 13 or more variables per particle). These runs typically require 15-30 wall-clock hours on 120 processors of Hopper. With the Maxwell (EM) field model, tests show good scaling at fixed problem size (512^3 cells) to 50,000 processors (see Figure below).



We project the need for four classes of Warp runs: (1) Ensemble runs to optimize the output beam from the NDCX-II accelerator, for each class of target being shot. So far, we haven't used gradient methods for optimization because of particle noise; we hope to overcome this with larger runs. (2) Simulations of plasma injection into the drift-compression line and final-focus solenoid, which can be quite costly because the plasma flow is relatively slow (~10 ms) and it is necessary to operate on an electron timescale. Both EM and explicit electrostatic (ES) models are used; run times are comparable because the time-step size in the ES model, which is set by the need to resolve plasma oscillations, is often near the Courant limit for light waves on the mesh. Also, the EM algorithm scales more readily to very large numbers of processors. (3) Integrated simulations of one or more beams compressing in a neutralizing plasma (with properties obtained from plasma-injection runs as described above, or via measurements). Such runs require less computer time than plasma injection runs, because the beam is in the system for < 1 ms; however, ensembles are typically needed. (4) Detailed simulations resolving short time- and space-scales for, *e.g.*, two-stream instability. Since the highest growth rates for a cold beam and plasma are for short wavelengths, while we seek to capture the overall system scale, such runs can be costly, even in axisymmetric (*r,z*) geometry. Projected 3-D runs will, of course, require substantially greater resources.

1.4.2 Data and I/O

Scratch (temporary) space: few TB
Permanent (can be shared, NERSC Global Filesystem /project): 1 TB (including Home)
HPSS permanent archival storage: 40 TB
I/O Rates: we have only been bandwidth limited in the sense that loading the code onto many processors has been an issue, requiring special approaches (e.g., a static build of Python) that we would refer to avoid if possible.
All of these increased requirements derive from the greater problem sizes anticipated, and from a somewhat greater number if runs (factor of several)

1.4.3 Scientific Achievements with 32X Current Resources

Our projected increase in needs, as described above, is by somewhat more than a factor of 32. Thus the achievements described above are relevant here.

1.4.4 Parallelism

A typical run is expected to use tens of thousands of core, with large runs using hundreds of thousands. Ensemble runs (with many cases run concurrently in a singe submission) can, in principal, make good use of any number that are available to us.
We will definitely need multiple jobs running concurrently; up to 10,000 in an ensemble would be useful, but not strictly necessary. In non-ensemble cases, modest numbers of large runs are needed; that is, a similar total core count, but fewer simultaneous jobs..

1.4.5 Memory

Per-core, we anticipate needs comparable to those of current-day runs, that is, of order 1 GB per core (number per node will depend upon the number of cores per node). Aggregate memory will increase commensurately with the number of cores.

1.4.6 Many-Core and/or GPU Architectures

We do not currently employ GPU's, but have been exploring the possibility. While the central loop of Warp could be readily adapted, end-case challenges (diagnostics, boundary conditions, etc.) make a full port to a GPU architecture challenging. We will rely on NERSC assistance to help us assess what is possible here, and to implement the best strategy.

1.4.7 Software Applications and Tools

We anticipate need for MPI, and the capability for MPI at the Python level. We need compilers for Fortran, C, C++, and perhaps OpenACC, OpenCL or something similar for accelerator-enabled code. Much of our I/O in Warp (e.g., dump files) is through Python, posing challenges for performance; some code restructuring may be needed to take advantage of MPI I/O.

1.4.8 HPC Services

We anticipate a need for consulting as the available NERSC resources evolve. Much of our visualization is in-line; CGM files are likely to remain important to our workflow. We hope to be able to make increased use of VisIt with Warp. ALE-AMR currently uses VisIt as its primary visualization tool. Our needs with regard to other services are not unusual.

1.4.9 Time to Solution and Throughput

We do not anticipate any unusual needs beyond the existing pattern of queue structures, limits, etc.

1.4.10 Data Intensive Needs

Larger Warp runs may ultimately drive some use of data-intensive computing techniques. We do not anticipate any special needs associated with data-intensive computing.

1.4.11 What Else?

Interactivity (on the main NERSC computer, for parallel computations) is essential for debugging and diagnostics development, because Warp produces many of its diagnostics on-line (thereby avoiding massive data transfers, offloading mostly processed data).

The Warp code is deeply tied to Python. Dynamic libraries are part of the core of Python and continued support for them would be useful. Dynamic libraries are not actually essential, since Warp and Python can be built statically, however, static loading requires modification of the Python source, introduces a more complicated and fragile build process, and makes installing new and upgrading existing packages more difficult since they must be manually incorporated into the build system (when they could otherwise be installed independently). Solving the outstanding issue of scalable and efficient loading of dynamic libraries would remove the need for the static build and would greatly ease code maintenance and preserve flexibility.

We will continue to have a need for scatter-add deposition of source terms from the particles onto the mesh. Integrity of these terms must be preserved as we optimize performance using (probably) MPI plus on-node parallelism, and AMR.

1.4.12 Requirements Summary Worksheet

(Please note that this table aggregates expected needs for Warp, ALE-AMR, and other beam and hydrodynamics codes)

	Used at NERSC in 2012	Needed at NERSC in 2017
Computational Hours (Hopper core-hour equivalent)	0.6 Million	40 Million
Scratch storage and bandwidth	0.3 TB	3 TB
	(not critical)	(not critical)
Shared global storage and bandwidth (/project)	0.1 TB	1TB
	(not critical)	(not critical)
Archival storage and bandwidth (HPSS)	1 TB	40 TB
	(not critical)	(not critical)
Number of conventional cores used for production runs	3000	100,000
Memory per node	1 GB	1 GB
Aggregate memory	3 TB	100 TB