Planning Simulations

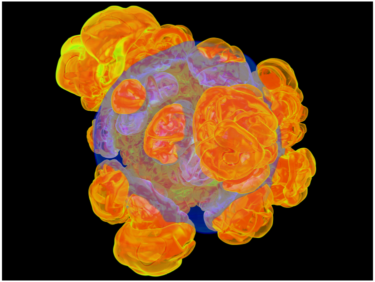
Dean Townsley
University of Alabama

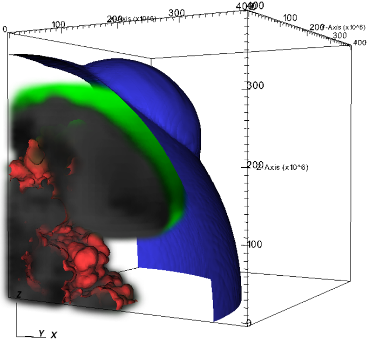
- Planning and budgeting simulations
- Planning for campaign execution

My perspective and working example

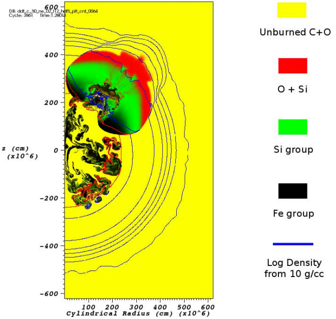
Reactive hydrodynamics simulations of thermonuclear powered supernovae
i.e. exploding compact stars

Large simulations required to capture turbulent acceleration of nuclear flame.
Need scale range for turbulence cascade.



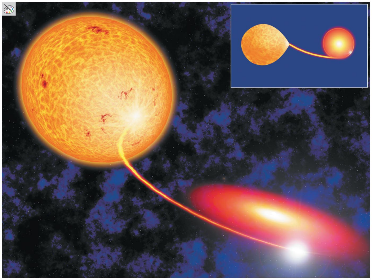


Much Iron-group made in Thermonuclear Supernovae



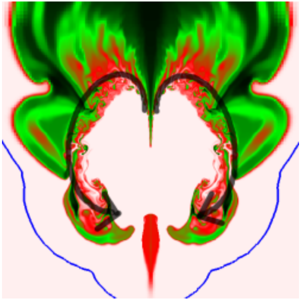
Most Iron-group material (Nickel, Iron) are produced in thermonuclear-powered supernovae (also called white dwarf supernovae)

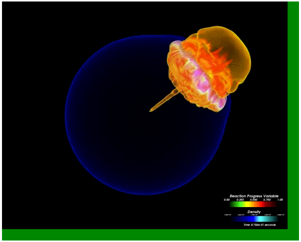
Take longer to happen – shows in interstellar gas



Budget Planning for Large Simulations

- Need to have a pilot – simulation or calculation that costs much less than real simulation but from which you can compute a full simulation's cost.
- Good examples:
  - 2D simulations
  - lower resolution 3D simulations





Need to understand your scaling

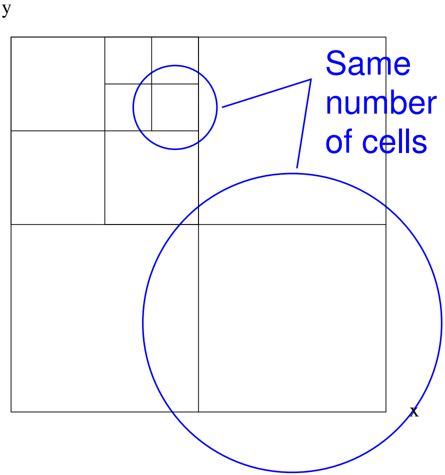
- Weak Scaling - a larger problem on a larger machine - often a given
- Strong Scaling - same problem on a larger machine - generally limited

Step back a bit:

First step: need a unit of work
then can quantify how long it takes

Unit of work: the block-step

For domain-decomposed fluid simulations



- Space divided into regions and sub-regions
- Each divided into some grid of cells (e.g. $16 \times 16 \times 16$)
- Good work unit: time to compute new fluid state for a "block" cells
- i.e. a block-step

Features of the block-step

- Larger version of cell-step time
- Is just a time: e.g. $t_b = 200$ milliseconds

Weak scaling is built in / assumed
i.e. for any problem size:

$$\text{execution time} = (\#\text{ of blocks}) \times (\#\text{ steps}) \times t_b$$

This is the necessary property of the work unit:
simulation cost can be expressed via a weak scaling formula like this; the work unit is the cost unit

Other units

What is the work unit for your application?

Strong scaling

Weak scaling is built in

t_b is independent of problem size assuming large load

Strong scaling:

is cost for computing a work unit independent of processor load?

Ideal: size of cluster doesn't matter

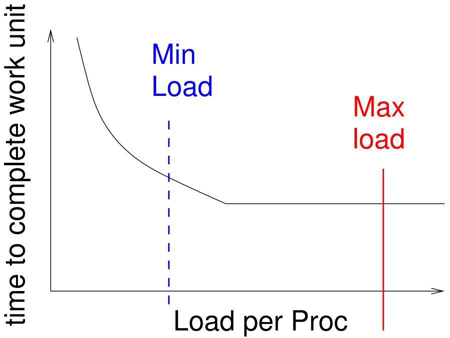
Note "processor" here is abstract, could be node, etc

Scaling and Load

Limits to scaling appear as increase in cost of computing work unit, t_b, at small processor load

Limit at large load usually memory

Plan simulations to give load in strong scaling region Separation between load limits gives your "flexibility"



Budgeting Simulations

Simulations should

- Target between your load margins
- Meet science goals in specific metric
  (scale contrast for turbulence, resolution of gradients)
- Cover applicable parameter space

Budget for resolution study
typically not necessary for all of parameter space
multidimensional methods assume you are doing this

Planning for Campaign Execution

some infrastructure suggestions

Regression Testing

version freeze would be nice, but...

In addition to component and regular regression testing

- Regression tests for specific science problem setup
- Split into separate stages to cover all epochs of science simulations
- Include performance testing
- Both short (automated) tests and longer tests driven by science goals

Job Management

Simulations will run over many job submissions

Select a partition size and chain jobs with dependency.

Allows scheduler (human or algorithm) to manage better

- Queue script checks status of previous job and sets up
- Use load margins to target partition size
- stable job size may be worth some science planning (e.g. refinement pattern)

Monitoring Run Health

Run health monitoring is essential at extreme scales – stakes are high

What to include?

- Summary variables
- As close to science outcomes as possible, and more
- Cumulative cost function (in block-steps) - monitors performance
- snapshot / slices / movies - low-cost, scripted vis

Simplicity is fine - just some scripts - not too complex to use (just some scripts generating web pages with plots)

Example for supernova simulation

After each job, new movie frames are generated and plots updated

Monitoring Run Health

Reported values can extend beyond science data
  e.g. boundary condition behavior, conservation metrics

Comparisons to reference runs or pilot runs on summary plots very useful

Cop-pilot cross-check

For expensive runs, can be helpful to have someone cross-check your parameter files.
This is well-known to reduce operator errors

Versioning and Recording Versions

Others will talk about more extensive practice, but even simple versioning is essential

Simple can go a long way, but discipline is essential.
Like keeping a lab book - an essential activity.

- Choose a way to version control - something you understand
- Use project branches to reduce barrier to commits
- Never run something that is not committed (well try)
- Operator notes, "runwith" notes - plan a standard place
- Use submission scripts (qsub.sh) to record submission parameters
- Don't re-use run directories (okay for cheap runs, not expensive ones)

Planning for the Unplanned

Important to plan for possible job problems

- Balance output time to not impact simulation time
- Output more checkpoints than you need and delete extras
- May not immediately know if simulation is "okay"
- Trade-off between data retained and data that can be regenerated
- Visualization may require higher cadence

Summary

Budgeting Simulations

- Use work unit appropriate to your application
- understand your application's scaling limits
- Target simulations at load range for good scaling

Preparing for Simulation Campaigns

- Test your science setup and be prepared for changes
- Do parallel analysis and monitor run health
- Diligently track what is used for your science runs