

The slide features a decorative design with orange lines and a graphic. A thin orange line forms an L-shape in the upper left, and another forms an L-shape in the lower right. In the bottom left corner, there is a graphic of three overlapping, semi-transparent bell curves in shades of brown and orange, with the tallest curve on the left and the others decreasing in height and shifting to the right.

Rickshaw

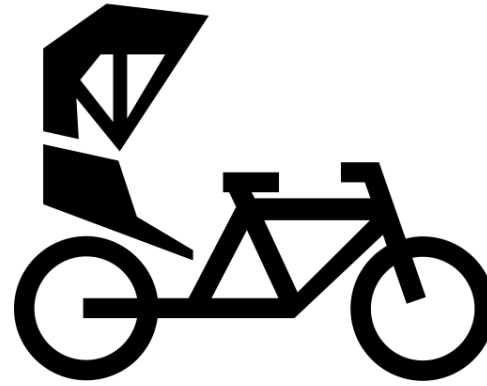
Technical Workshop on Fuel Cycle
Simulation, July 20th, 2017

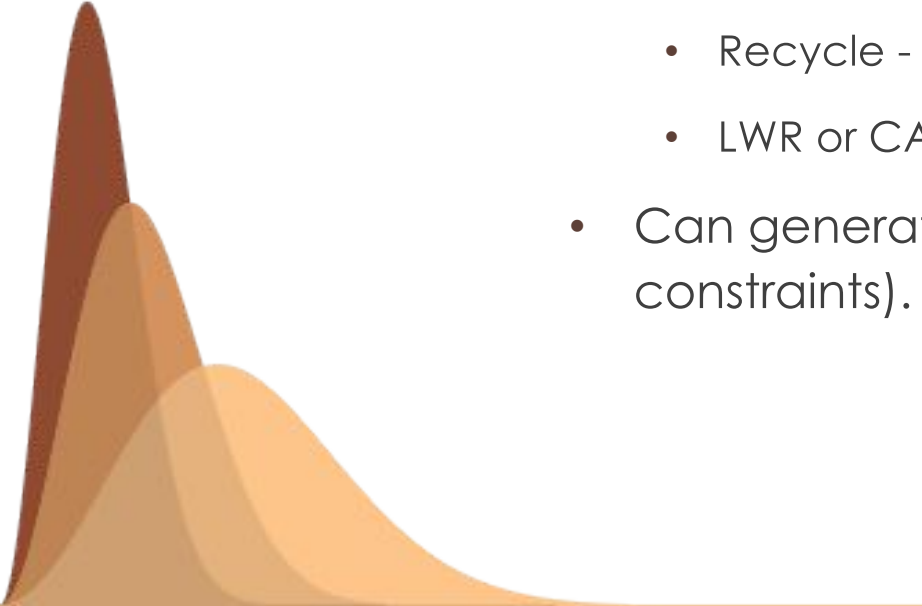
Columbia, South Carolina

Robert Flanagan, Anthony Scopatz

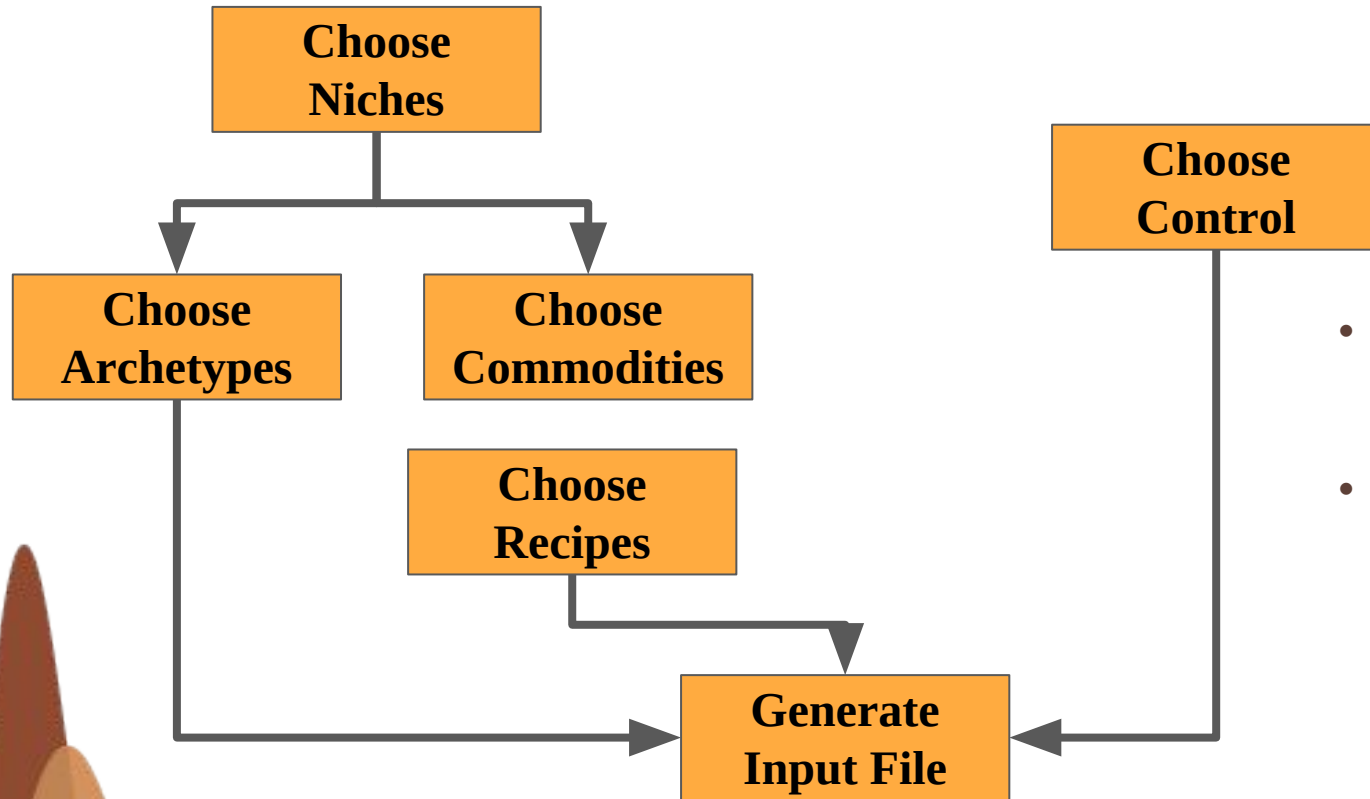


Rickshaw



- Rickshaw is a Python-powered stochastic driver for Cyclus. Rickshaw will create randomly-generated, fully-valid Cyclus input files
 - These simulations contain a variety of 'niches' to allow for different types of fuel cycles.
 - Once through
 - Recycle - Single Pass or Multiple
 - LWR or CANDU
 - Can generate, and run, any number of Cyclus simulations (within machine constraints).
- 

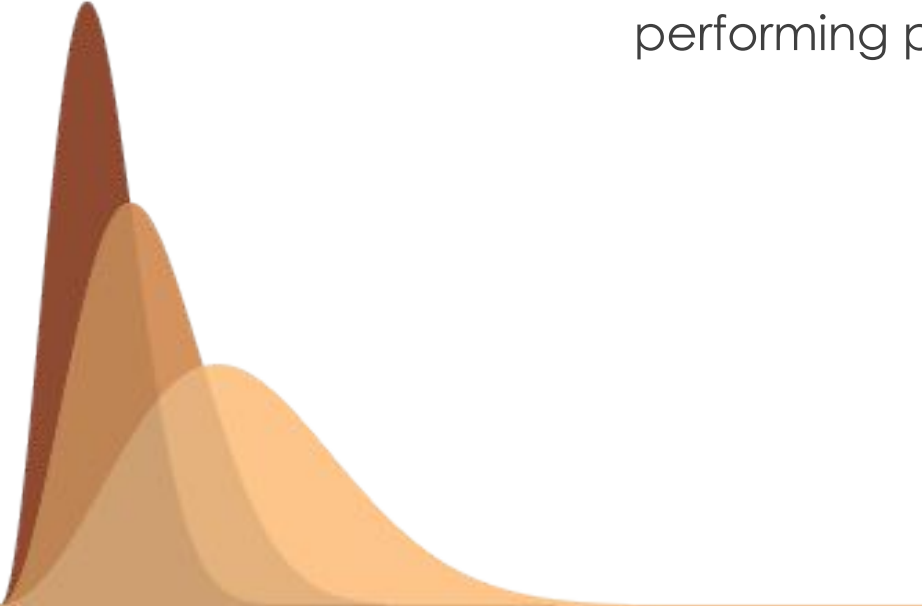
How?



- First, Rickshaw chooses the niches it will use for a simulation.
 - Niches cover types of fuel cycle facilities; mine, reactor, enrichment, etc.
 - Each niche can have several facilities within it; for example reactor might contain - LWR, HWR, etc.
- Rickshaw then chooses a facility within each niche.
- Once this backbone for the simulation is created, Rickshaw goes through each facility, pings Cyclus to get the input for that facility, and randomly generates values for required fields.
- Recipes are also randomized by Rickshaw. Facilities have possible recipe templates attach to them. The mass fractions of recipes are generated randomly.

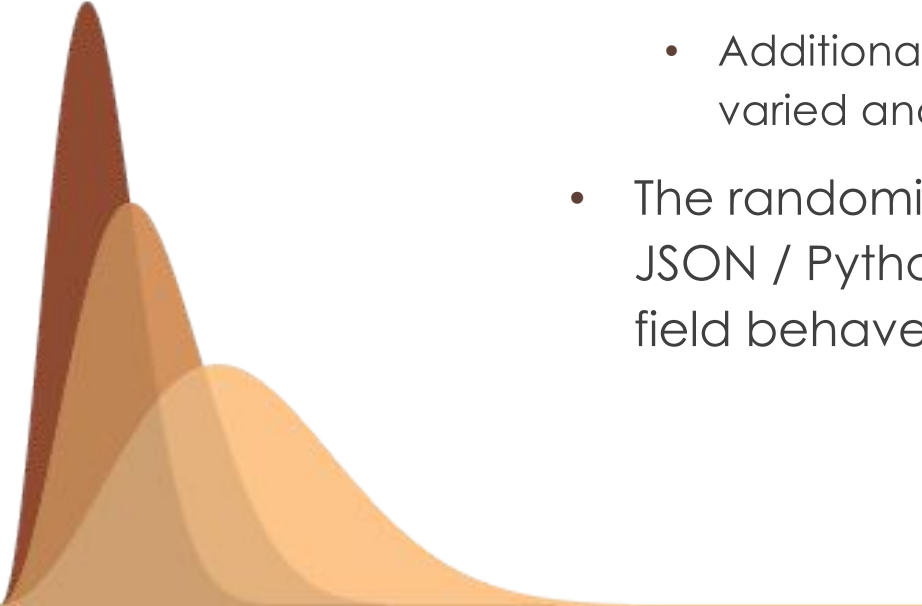


Why?

- To model Cyclus simulations as a Gaussian process.
 - If we run sufficient simulations, we can break down a scenario into a set of hyperparameters, and therefore predict results without running any further Cyclus simulations.
 - Furthermore, the Cyclus ecosystem was lacking a standardized method for performing parameter sweeps.
- 





Template Mode

- As mentioned, Rickshaw can also be used as a tool for more defined simulations.
 - This is done by feeding Rickshaw with a input file containing the features that need to be fixed.
 - Any and all parts of a simulation can be fixed; niches, facilities in each niche, etc.
 - Additionally a full cyclus input can be passed to Rickshaw with parameters to be varied and how to vary them and Rickshaw will only vary those parameters.
 - The randomization can be modified as well. Rickshaw accepts Jinja2 / JSON / Python templating that allows you to choose how a randomized field behaves.
- 

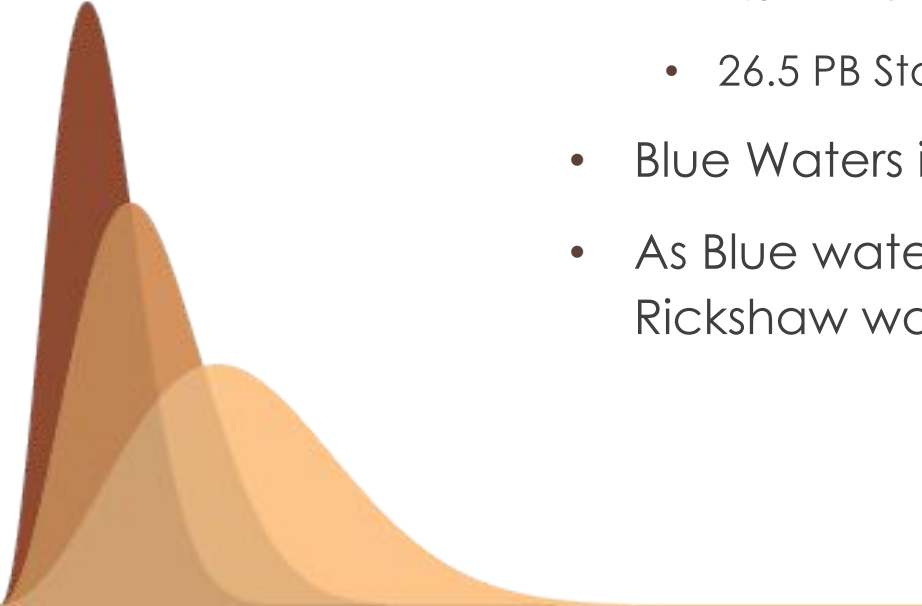


Large Scale

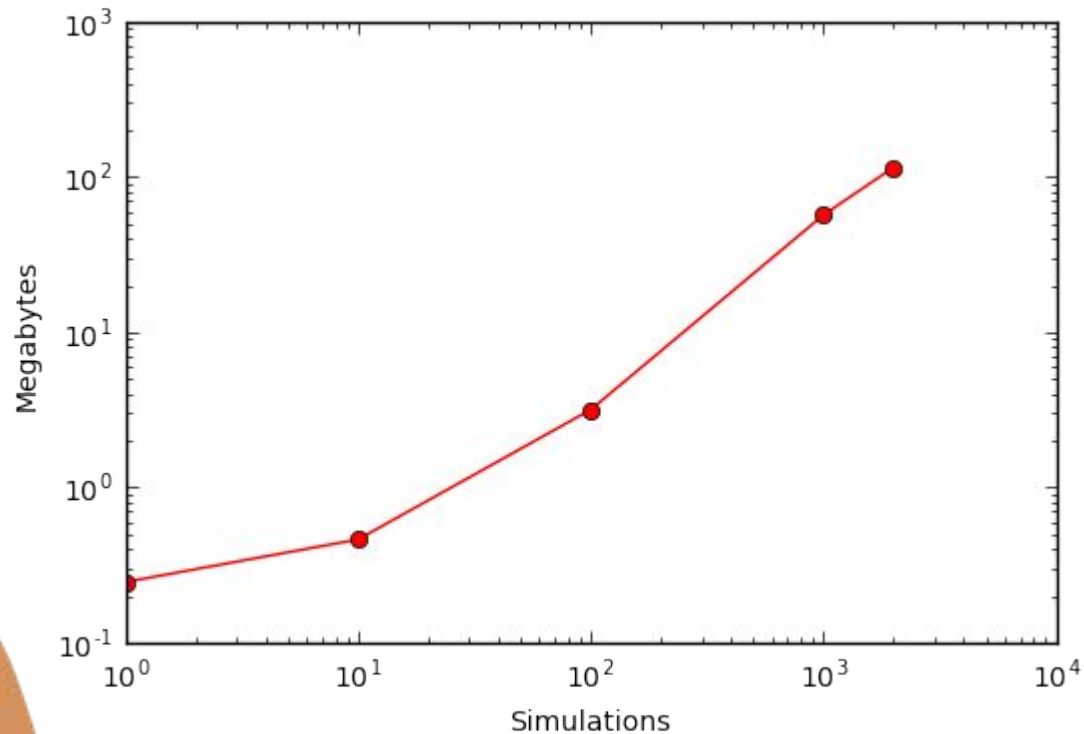
- Generating 100,000 input files for Cyclus is wonderful, but now they all need to be run or we're just spinning our wheels.
 - On a person machine, this is going to take on the order of days.
 - Deploying these jobs to a compute cluster is more ideal.
 - Rickshaw currently does this automatically through two methods.
 - Docker
 - Blue Waters
 - Using Docker, Rickshaw attaches to a Docker manager and spawns the jobs across a Docker Swarm.
 - This is done on our local cluster – Meeseeks (~32 cores).
 - Meeseeks can run over 1 million simulations in a week.
- 



Larger Scale - Blue Waters

- In conjunction with University of Illinois at Urbana-Champaign, Rickshaw has been set up to work with Blue Waters.
 - Blue Waters is an NSF supercomputer.
 - 49,000 cores (1 petaFlop)
 - 1.5PB Memory
 - 26.5 PB Storage
 - Blue Waters is significantly larger than Meeseeks ~ 10,000x larger
 - As Blue waters doesn't use Docker swarm, a special run method of Rickshaw was built to specifically allow for this interaction.
- 

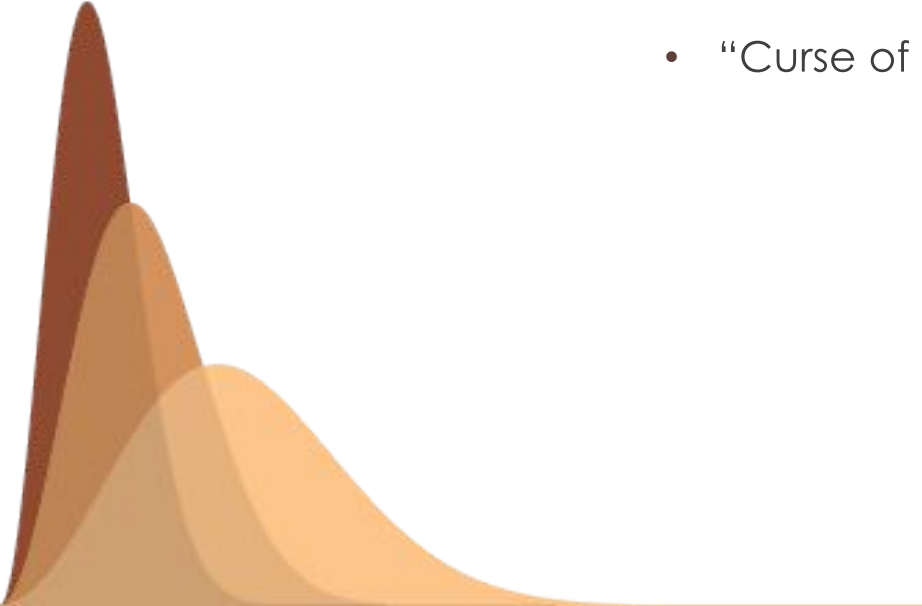
Troubles




- Namely, data...
- A single, low-fidelity Cyclus simulation output database can be on the order of 240 kb.
- Putting multiple simulations into a single data base helps to reduce the data sizes.
 - 1 Simulation – 240 KB
 - 10 Simulations – 460 KB
 - 100 Simulations – 3.1 MB
 - 2,000 Simulations – 57 MB
 - 1,000,000 Simulations – 57 GB
- Rickshaw's original goal was to eventually reach 1 billion simulations. If trends continue that would generate 57 TB of data.



Troubles

- Analysis and visualization of such large datasets is difficult.
 - Choosing what parameters to look at and which objective functions make sense for a broad range of fuel cycles.
 - Varying all of the input fields within even a simple Cyclus simulation could result in a parameter space of 10s to 100s.
 - “Curse of dimensionality”
- 



Future Work - Deployment

- Using Rickshaw as a stochastic deployment generator.
 - Rickshaw utilizes the deploy institution in Cyclus to deploy facilities at random through a simulation lifetime.
 - By default all facility prototypes can be deployed in random numbers at each time step in the simulation.
 - This behavior is largely for the stochastic approach of running 1B simulations.
 - What if you'd like to instead investigate a transition scenario?
 - Decreasing chance of deploying an LWR until phase out at some cutoff timestep.
 - Increasing chance of deploying a FR starting at some timestep.
- 