WORKFLOW AUTOMATION TOOLS FOR MANY-TASK COMPUTING

Geoffrey Lentner, Lead Research Data Scientist

PURDUE UNIVERSITY | Rosen Center for Advanced Computing

WORKFLOW AUTOMATION TOOLS FOR MANY-TASK COMPUTING

Purdue Community Workshop

This workshop outlines the paradigm of *many-task* computing and covers different scenarios and tools to manage such workflows.

An introduction to ***hyper-shell*** is included at the end.

Geoffrey Lentner
Lead Research Data Scientist
Rosen Center for Advanced Computing
Purdue University

PURDUE UNIVERSITY | Rosen Center for Advanced Computing

Preface

About Us

The ***Rosen Center for Advanced Computing*** (RCAC) provides advanced computational resources and services to support Purdue faculty and staff researchers. We conduct our own research and development to enhance the capabilities of these resources, as well as provide expertise in a broad range of HPC matters and activities.

This includes providing workshops on common topics surrounding the use of HPC systems.

Preface

About Me

Members of the *Scientific Applications* team have a number of responsibilities; including,

- System configuration and services
- Scientific software
- Advanced projects and consulting
- Teaching
- Outreach and Engagement
- Innovation

Preface

About You

Prerequisites. We assume you have some prior experience working on a Linux cluster, using the scheduler and simple command-line programs.

- Linux command-line basics (Unix 101 and Unix 102)
- Cluster basics (Clusters 101)

Preface

Outline

- What do we mean by many-task computing?
- Why not just use the scheduler?
- What tools are available to manage tasks?
- What is hyper-shell and how to use it.

1

Introduction

What are we even talking about?
What is many-task computing?

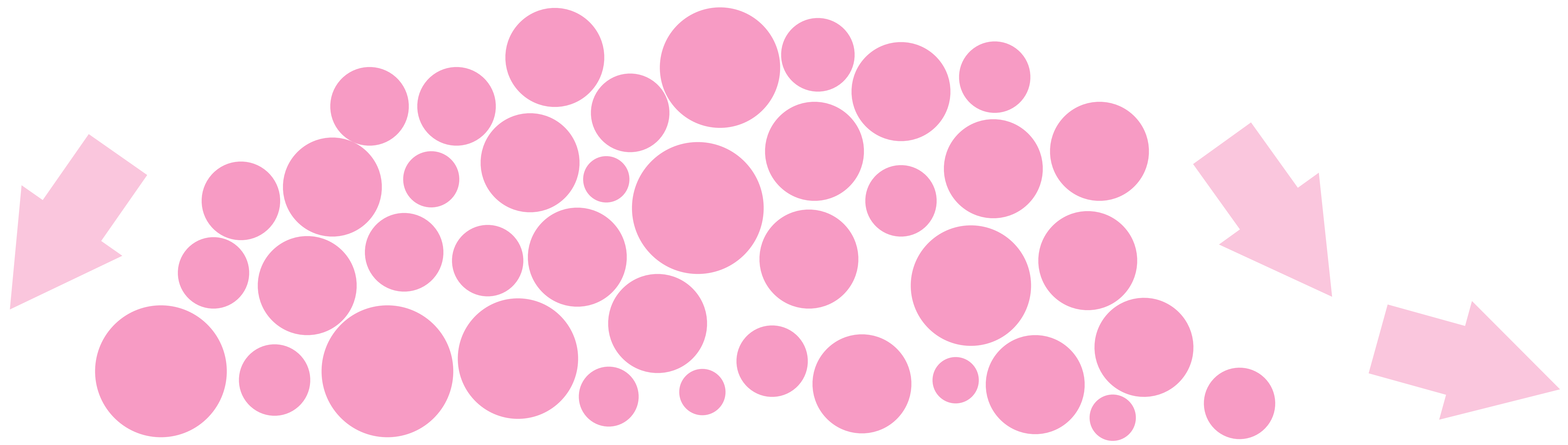
Introduction

What?

High-Throughput, *Many-Task* Computing

Large number of Independent Tasks without Communication

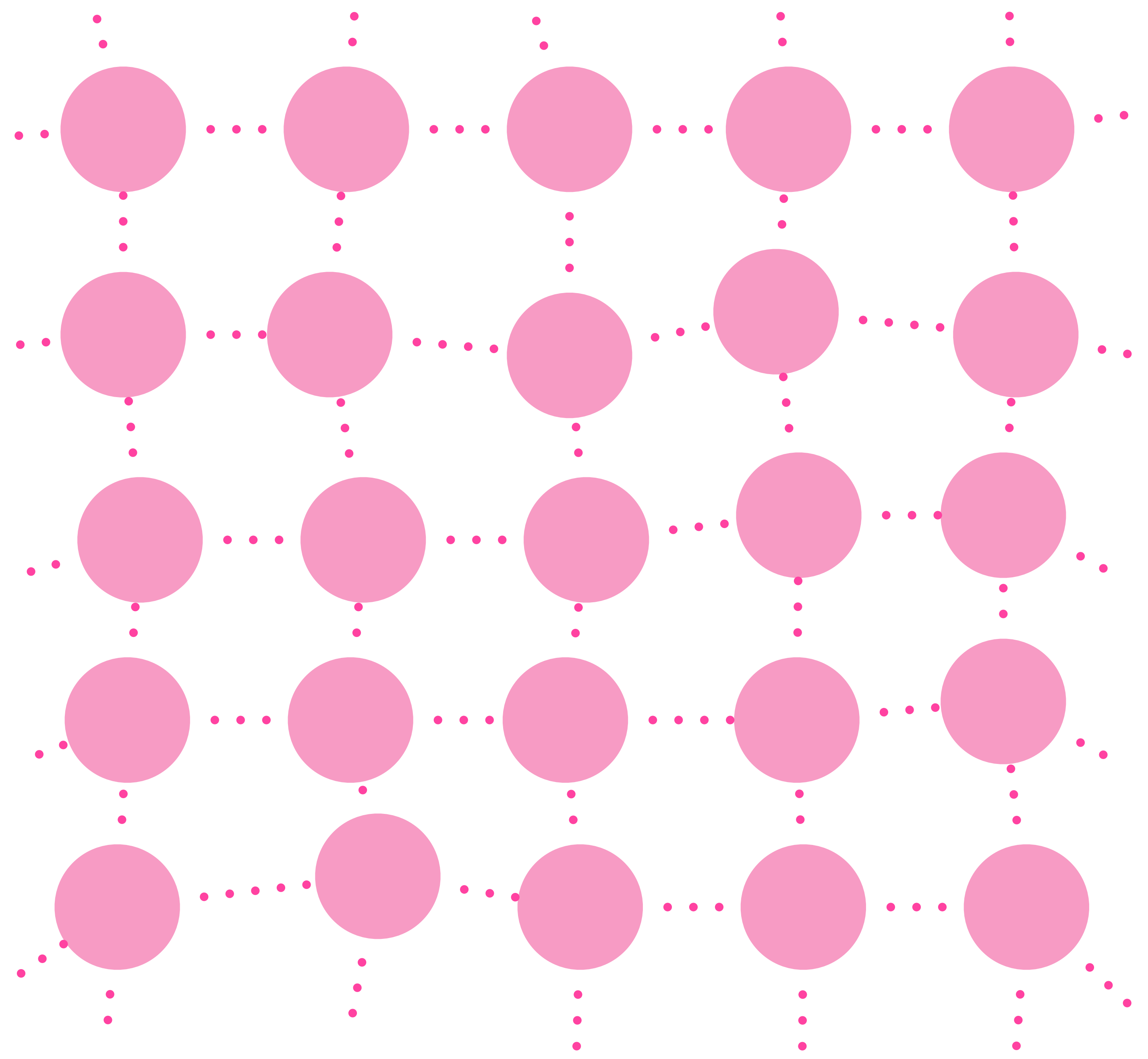
Defined by the number of tasks completed or volume of data processed. Elements of the workflow are entirely independent and may run anywhere, even across administrative boundaries.



High-Performance Computing

Coherent Coupled Tasks with Communication

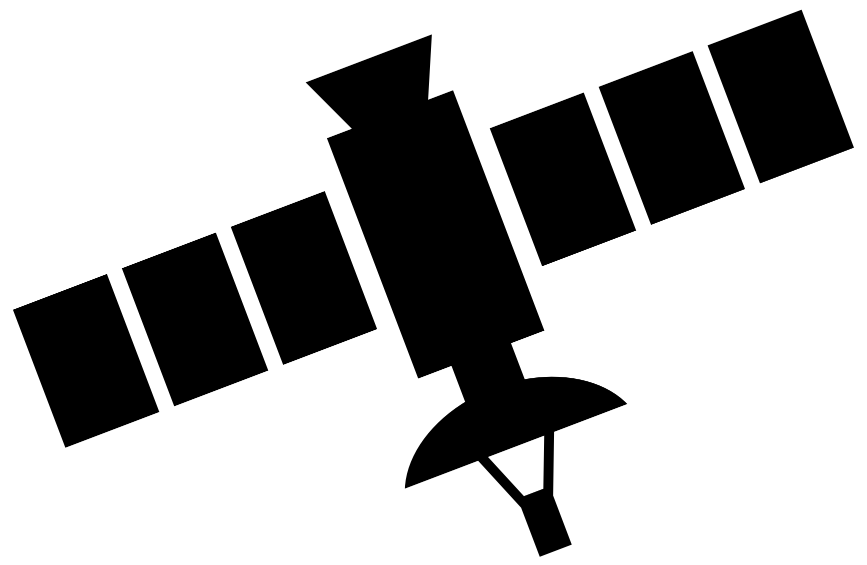
Monolithic simulation where tasks run simultaneously and define some physical or abstract space, working together to solve one problem.



Introduction

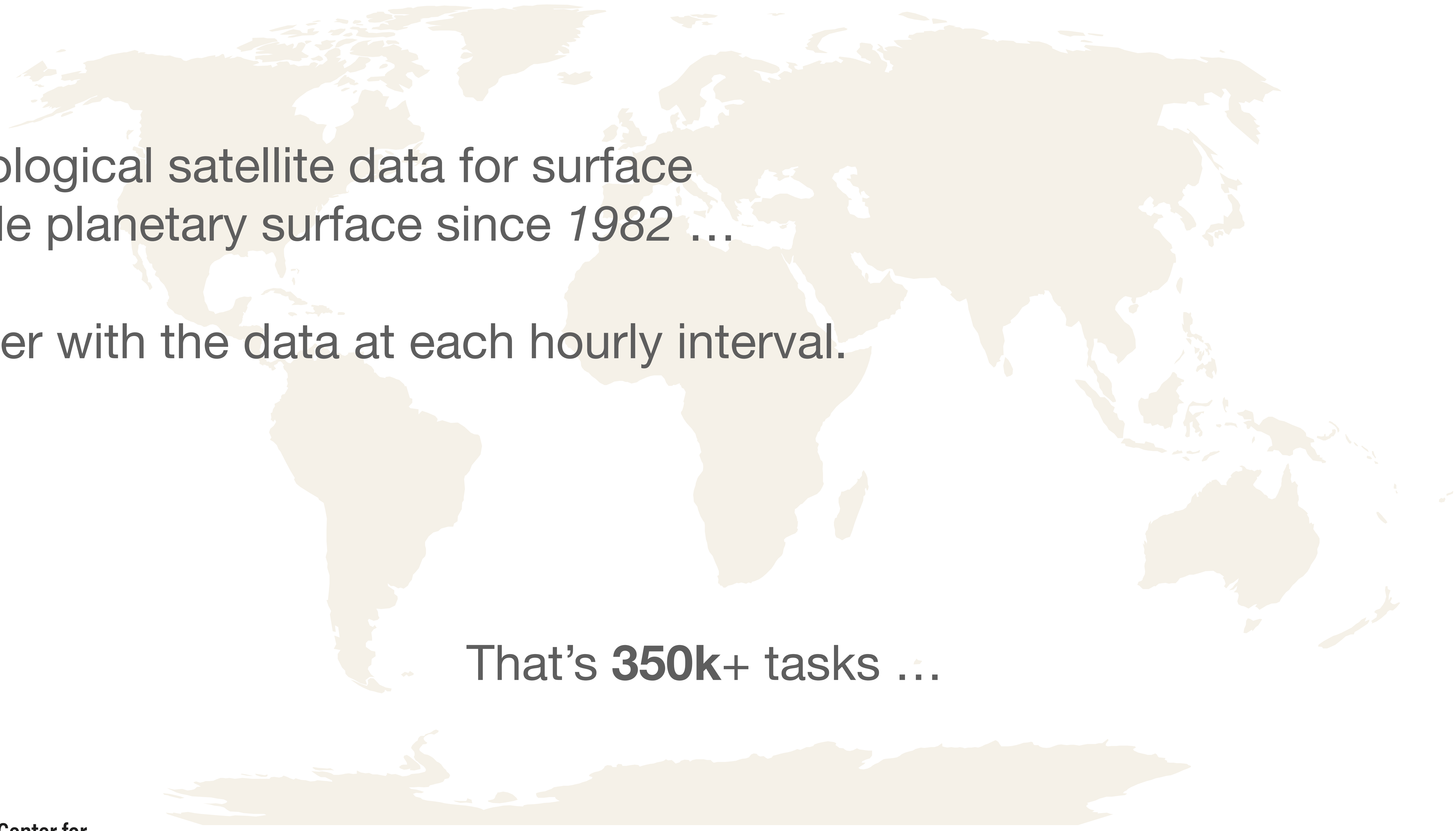
Use cases

- Data Processing and Analysis
- Machine-Learning Experiments (model tuning)
- Bioinformatics Tasks
- Parameter Searchers (calculations)



Introduction

Example 1



Process *hourly* meteorological satellite data for surface parameters for the whole planetary surface since *1982* …

Compute new parameter with the data at each hourly interval.

That's **350k**+ tasks …

Rosen Center for Advanced Computing

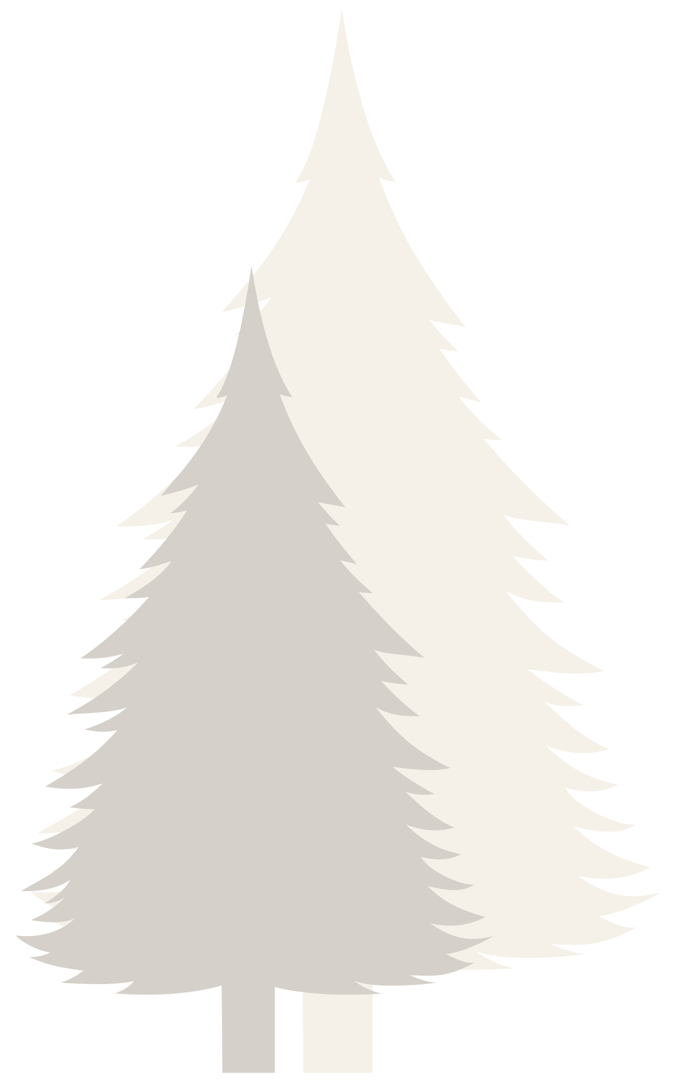
Introduction

Example 2

Train a *stochastic* model against decades of forest survey data to predict ***tree species*** population change …

For many species, for many regions, ***hundreds of times*** …

That's **200k+** tasks …



Purdue University | Rosen Center for Advanced Computing

Introduction

Example 3

Calculate some mathematical coefficient by searching a *higher*-dimensional parameter space …

That's **1M**+ tasks …

Introduction

Why not manage tasks directly?

People tend to just iterate through tasks inside the same application language where the computation is defined; e.g., *Python, Julia, R, MATLAB, Mathematica, …*

… because it's simple, obvious, and works.

… until it doesn't.

PURDUE UNIVERSITY | Rosen Center for Advanced Computing

Introduction

Where does this break down?

Introduction

“ *I got an allocation on the computing cluster and my **MATLAB** workflow runs okay …*

*… but now I have **six** nodes and struggling to make distributed computing work.*

Introduction

Building ***HPC-friendly*** workflows means writing less code …

Introduction

Problems

A direct approach within the same script …

- *is monolithic (what if there's an error),*
- *is inflexible (couples orchestration with problem code),*
- *a challenge for novice programmers, and*
- *tedious.*

2

Use the scheduler

Why not just use SLURM?
What are the benefits and limitations?

Using the scheduler

In the ideal case

SLURM (or equivalent) is actually the best approach.
It is literally the scheduler's responsibility to manage tons of independent jobs.

`jobscript.sh`

```bash
#!/bin/bash
#SBATCH -N1 -n1 -t 00:30:00
#SBATCH ...

# load software
module load matlab/R2021A

# execute single instance
matlab -nosplash -singleCompThread -batch "..."
```

Control the varying inputs somehow, such as with an environment variable, or substitution …

Using the scheduler

Job Arrays

Most schedulers support “job arrays” in which the same job is executed some large number of times with only environment variables being different.

job-array.sh

```
#!/bin/bash
#SBATCH -N1 -n1 -t 00:30:00
#SBATCH -a 1-999
#SBATCH -o task-%A.%a.out
#SBATCH -e task-%A.%a.err

# load software
module load matlab/R2021A

# execute single instance
matlab -nosplash -singleCompThread -batch “...($SLURM_ARRAY_TASK_ID)”
```

Just the variable we were looking for…

Using the scheduler

So why would you use some other **“workflow automation”** tool?

Using the scheduler

Practical limits

Site administrators don’t want users …

- *submitting millions of jobs,*
- *filling up the database,*
- *impacting site-wide throughput, and*
- *polluting the queue.*

Using the scheduler

Submit a **“pilot job”** *instead …*

3

Workflow Automation Tools

What other tools are available?
What are the benefits and limitations?

Workflow Automation Tools

Create a task file

We could ***source*** this file and it would be valid, running the commands *one at a time* …

task.in

```
matlab -nosplash -singleCompThread ... -batch "...(1, ...)"
matlab -nosplash -singleCompThread ... -batch "...(2, ...)"
matlab -nosplash -singleCompThread ... -batch "...(3, ...)"
matlab -nosplash -singleCompThread ... -batch "...(4, ...)"
matlab -nosplash -singleCompThread ... -batch "...(5, ...)"
matlab -nosplash -singleCompThread ... -batch "...(6, ...)"
matlab -nosplash -singleCompThread ... -batch "...(7, ...)"
matlab -nosplash -singleCompThread ... -batch "...(8, ...)"
matlab -nosplash -singleCompThread ... -batch "...(9, ...)"
matlab -nosplash -singleCompThread ... -batch "...(10, ...)"
matlab -nosplash -singleCompThread ... -batch "...(11, ...)"
matlab -nosplash -singleCompThread ... -batch "...(12, ...)"
matlab -nosplash -singleCompThread ... -batch "...(13, ...)"
matlab -nosplash -singleCompThread ... -batch "...(14, ...)"
matlab -nosplash -singleCompThread ... -batch "...(15, ...)"
matlab -nosplash -singleCompThread ... -batch "...(16, ...)"
matlab -nosplash -singleCompThread ... -batch "...(17, ...)"
matlab -nosplash -singleCompThread ... -batch "...(18, ...)"
matlab -nosplash -singleCompThread ... -batch "...(19, ...)"
matlab -nosplash -singleCompThread ... -batch "...(20, ...)"
matlab -nosplash -singleCompThread ... -batch "...(21, ...)"
matlab -nosplash -singleCompThread ... -batch "...(22, ...)"
matlab -nosplash -singleCompThread ... -batch "...(23, ...)"
matlab -nosplash -singleCompThread ... -batch "...(24, ...)"
matlab -nosplash -singleCompThread ... -batch "...(25, ...)"
...
```

Workflow Automation Tools

Pass the file to a workflow automation tool

All of the utilities discussed here have a similar usage pattern

pilot-job.sh

```bash
#!/bin/bash
#SBATCH -N4 -t 1-00:00:00 --exclusive
#SBATCH ...

# load software
module load ???

# scale out tasks
<???> task.in ...
```

What can we put here?

Purdue University | Rosen Center for Advanced Computing

Workflow Automation Tools

xargs

Build and execute command lines from standard input

Pros	Cons	
Available by default	Low observability	Monolithic
Simple	Not Scalable (single node)	No retries
Templates	UNIX/Linux only	

Example

```
$ xargs -a task.in -d "\n" -P128 -I {} bash -c "{}"
```

Workflow Automation Tools

srun

Slurm launcher and job step manager

Pros	Cons	
Available	Low observability	Not Scalable (site limits)
Distributed	Linux/SLURM only	No retries
	Monolithic	

Example

```
$ cat task.in | xargs -d “\n” -P512 -I {} srun -n1 ... {}
```

Depending on how big you want to go this could be ***problematic*** for the *same reasons* submitting many jobs is.

https://slurm.schedmd.com/srun.html

Workflow Automation Tools

ParaFly

Execute tasks in parallel and report errors

Pros	Cons	
Simple	Low observability	No Templates
Retries / Failures	Not Scalable (single node)	
	UNIX/Linux only	
	Monolithic	

Example

```
$ ParaFly -c task.in -CPU 128 -failed_cmds task.failed
```

https://parafly.sourceforge.net/
https://github.com/ParaFly/ParaFly

Workflow Automation Tools

GNU Parallel

Execute tasks in parallel using one or more computers

Pros	Cons	
Usability	Low observability	I/O Limitations
Distributed	Not scalable (~10-20 nodes)	
*Retries / Failures	UNIX/Linux only	
Templates	Monolithic	

Example

```
$ cat task.in | parallel -j128 ...
```

https://www.gnu.org/

Workflow Automation Tools

Launcher

Scale out tasks on HPC systems

Pros	Cons	
Simple	Low observability	No Templates
Distributed	Linux only (HPC specific)	
Scalable	Monolithic	

Example

```
$ export LAUNCHER_WORKDIR=.
$ export LAUNCHER_JOB_FILE=task.in

$ ${LAUNCHER_DIR}/paramrun
```

https://github.com/TACC/launcher
https://portal.tacc.utexas.edu/software/launcher
https://www.tacc.utexas.edu/research-development/tacc-software/the-launcher

Workflow Automation Tools

hyper-shell

Process shell commands over a distributed, asynchronous queue

Pros		Cons
Retries	Usability	New
Flexibility	Scalability	
Cross-platform	Observability	
Persistent / management	Templates	

Example

```
$ hyper-shell cluster task.in -N128 --launcher=srun ...
```

https://github.com/glentner/hyper-shell
https://hyper-shell.readthedocs.io/

Workflow Automation Tools

Feature Comparison

	Distributed	Restart / Failures	Templates	Scalable	Observable	Cross Platform	Persistent
hyper-shell	✓	✓	✓	✓	✓	✓	✓
xargs			✓				
srun	✓			✓			
ParaFly		✓					
GNU Make		✓	✓				
GNU Parallel	✓	✓	✓		–		
*****Launcher***	✓	✓		✓	–		
******TaskFarmer***	✓	✓					

* https://www.tacc.utexas.edu/research-development/tacc-software/the-launcher
** https://docs.nersc.gov/jobs/workflow/taskfarmer/

Workflow Automation Tools

Use the tool that works for **you** *…*

… consider trying a new one if you need additional features.

4

hyper-shell

What is hyper-shell?
What problems is it trying to solve?
What are some of it’s features?

Common Usage

Example job script using *hyper-shell* on *Negishi*

pilot-job.sh

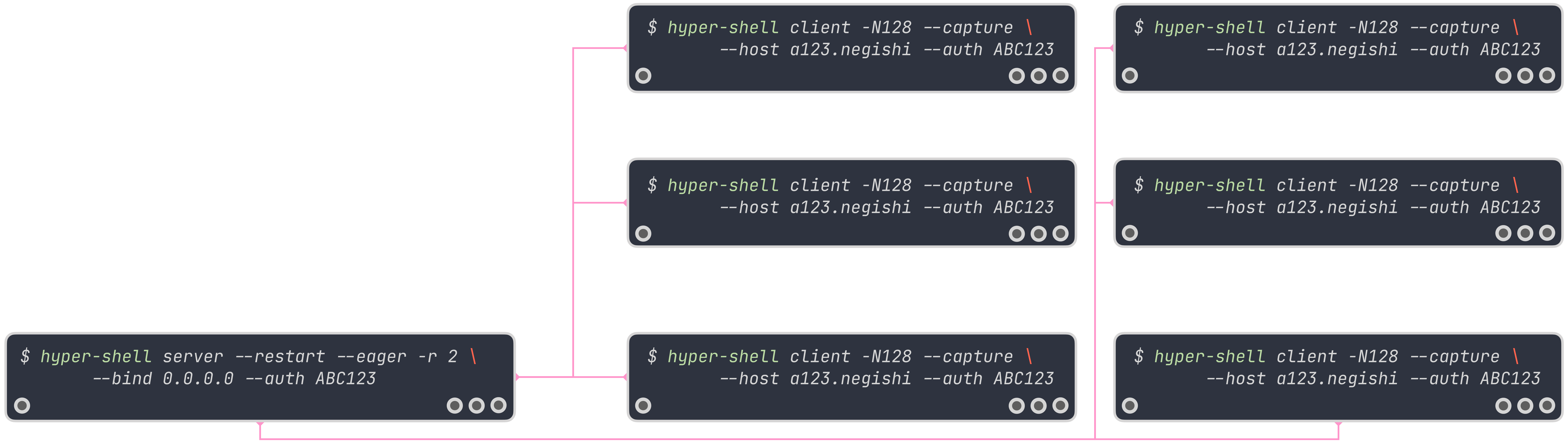
```
#!/bin/bash
#SBATCH -N4 -t 1-00:00:00 --exclusive
#SBATCH ...

# load software
module load hyper-shell

# launch workflow
hyper-shell cluster task.in -N128 --launcher=srun \
             --capture --failures=task.failed --no-db --no-confirm
```

Flexible (Not Monolithic)

Bring up the server first, then the clients later as needed



```
$ hyper-shell server --restart --eager -r 2 \
      --bind 0.0.0.0 --auth ABC123
```

```
$ hyper-shell client -N128 --capture \
      --host a123.negishi --auth ABC123
```

```
$ hyper-shell client -N128 --capture \
      --host a123.negishi --auth ABC123
```

```
$ hyper-shell client -N128 --capture \
      --host a123.negishi --auth ABC123
```

```
$ hyper-shell client -N128 --capture \
      --host a123.negishi --auth ABC123
```

```
$ hyper-shell client -N128 --capture \
      --host a123.negishi --auth ABC123
```

```
$ hyper-shell client -N128 --capture \
      --host a123.negishi --auth ABC123
```

Cross-Platform (Hybrid)

Clients do not need to run within the same platform environment

```
$ hyper-shell server --restart --eager -r 2 \
      --bind 0.0.0.0 --auth ABC123
```

```
C:\ hyper-shell client -N128 --capture \
        --host a123.negishi --auth ABC123
```

```
C:\ hyper-shell client -N128 --capture \
        --host a123.negishi --auth ABC123
```

```
C:\ hyper-shell client -N128 --capture \
        --host a123.negishi --auth ABC123
```

```
C:\ hyper-shell client -N128 --capture \
        --host a123.negishi --auth ABC123
```

```
C:\ hyper-shell client -N128 --capture \
        --host a123.negishi --auth ABC123
```

```
C:\ hyper-shell client -N128 --capture \
        --host a123.negishi --auth ABC123
```

Rich Templates

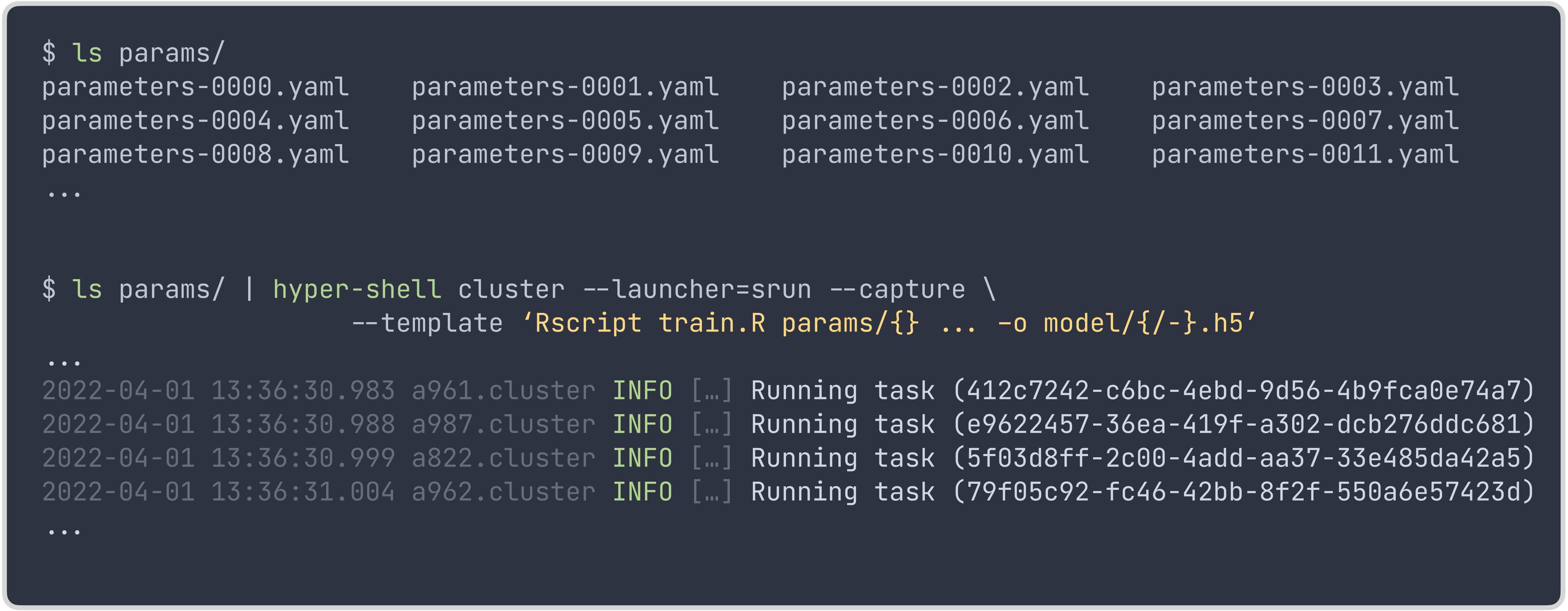
Dynamically build commands instead of reading from a static file

Filepath Operations
Shorthand notation provides convenient expressions, such as, {/-}, which expands to the base name without the file type extension.

Argument Slicing
Select items from the incoming argument with familiar notation, such as {[0]} for the first item, or {[:-2]} for the last two.

Shell Expansion
Interpolate the output of other programs into the command line, such as {% mktemp -d @ %}.

Lambda Expressions
Evaluate expressions (with Python), such as {= x * math.pi =}.



```
$ ls params/
parameters-0000.yaml    parameters-0001.yaml    parameters-0002.yaml    parameters-0003.yaml
parameters-0004.yaml    parameters-0005.yaml    parameters-0006.yaml    parameters-0007.yaml
parameters-0008.yaml    parameters-0009.yaml    parameters-0010.yaml    parameters-0011.yaml
...


$ ls params/ | hyper-shell cluster --launcher=srun --capture \
                    --template 'Rscript train.R params/{} ... -o model/{/-}.h5'
...
2022-04-01 13:36:30.983 a961.cluster INFO […] Running task (412c7242-c6bc-4ebd-9d56-4b9fca0e74a7)
2022-04-01 13:36:30.988 a987.cluster INFO […] Running task (e9622457-36ea-419f-a302-dcb276ddc681)
2022-04-01 13:36:30.999 a822.cluster INFO […] Running task (5f03d8ff-2c00-4add-aa37-33e485da42a5)
2022-04-01 13:36:31.004 a962.cluster INFO […] Running task (79f05c92-fc46-42bb-8f2f-550a6e57423d)
...
```

Persistent Metadata

A database is used to persistent and query task metadata

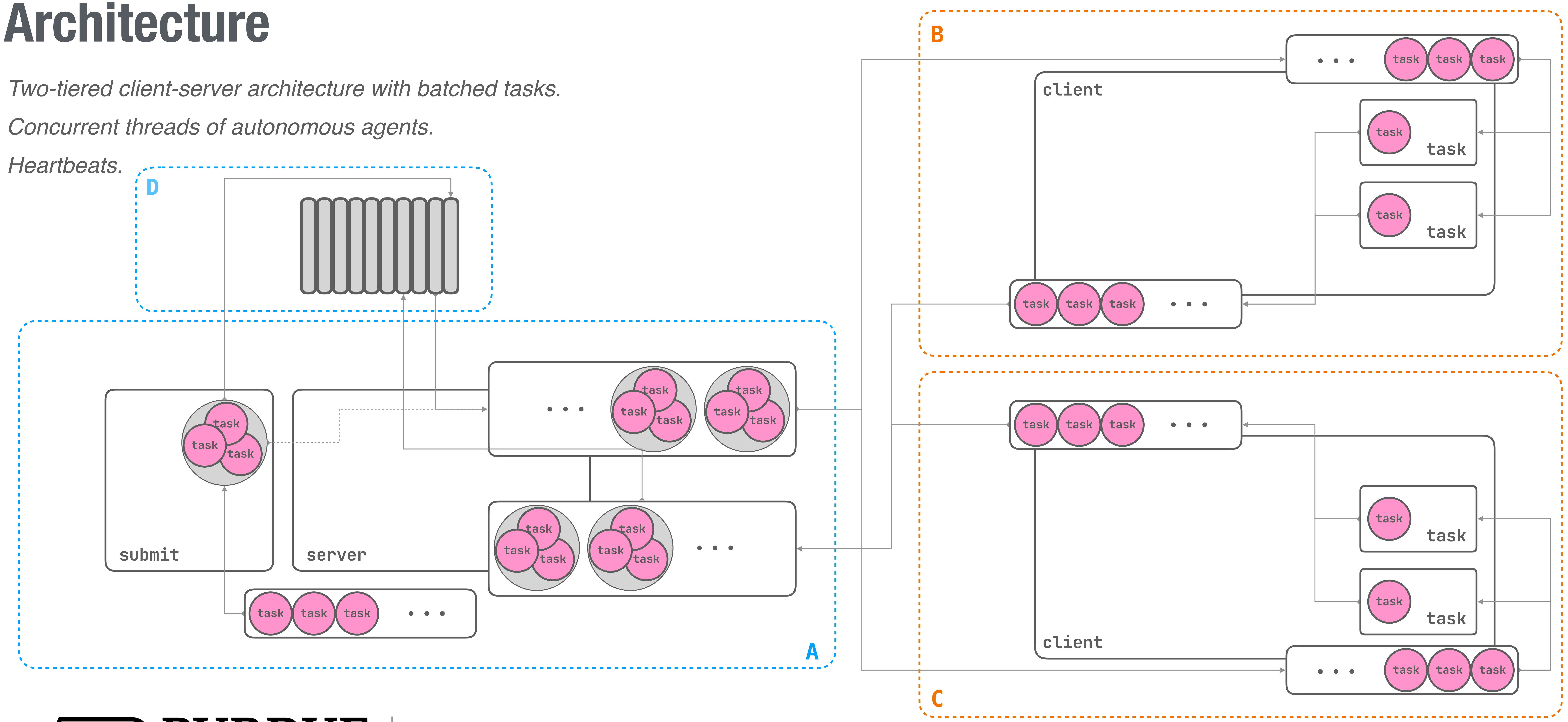
```
$ hyper-shell task search --where 'exit_status != 0' 'attempt == 2' --json
[
    {
        "id": "f05ceea8-3232-4b73-88e3-c26966865d68",
        "args": “matlab -nosplash -singleCompThread -batch ...”,
        ...
    },
    {
        "id": "351f748c-d1d8-44a7-a1ff-37b4fa64ffd8",
        "args": “matlab -nosplash -singleCompThread -batch ...”,
        ...
    }
]
```

Architecture

Two-tiered client-server architecture with batched tasks.

Concurrent threads of autonomous agents.

Heartbeats.



5

Alternatives

What other types of tools are out there?

Alternatives

Makefile

Alternative solutions using dependency-graph specific syntax

- ***GNU Make*** *(available, robust, dynamic, but not scalable)*,
  https://www.gnu.org/software/make/manual/make.html
- ***Snakemake*** *(couples execution with environment and orchestration)*,
  https://snakemake.readthedocs.io/en/stable/
- ***Makeflow*** *(couples execution with environment and orchestration)*
  https://cctools.readthedocs.io/en/latest/makeflow/

Alternatives

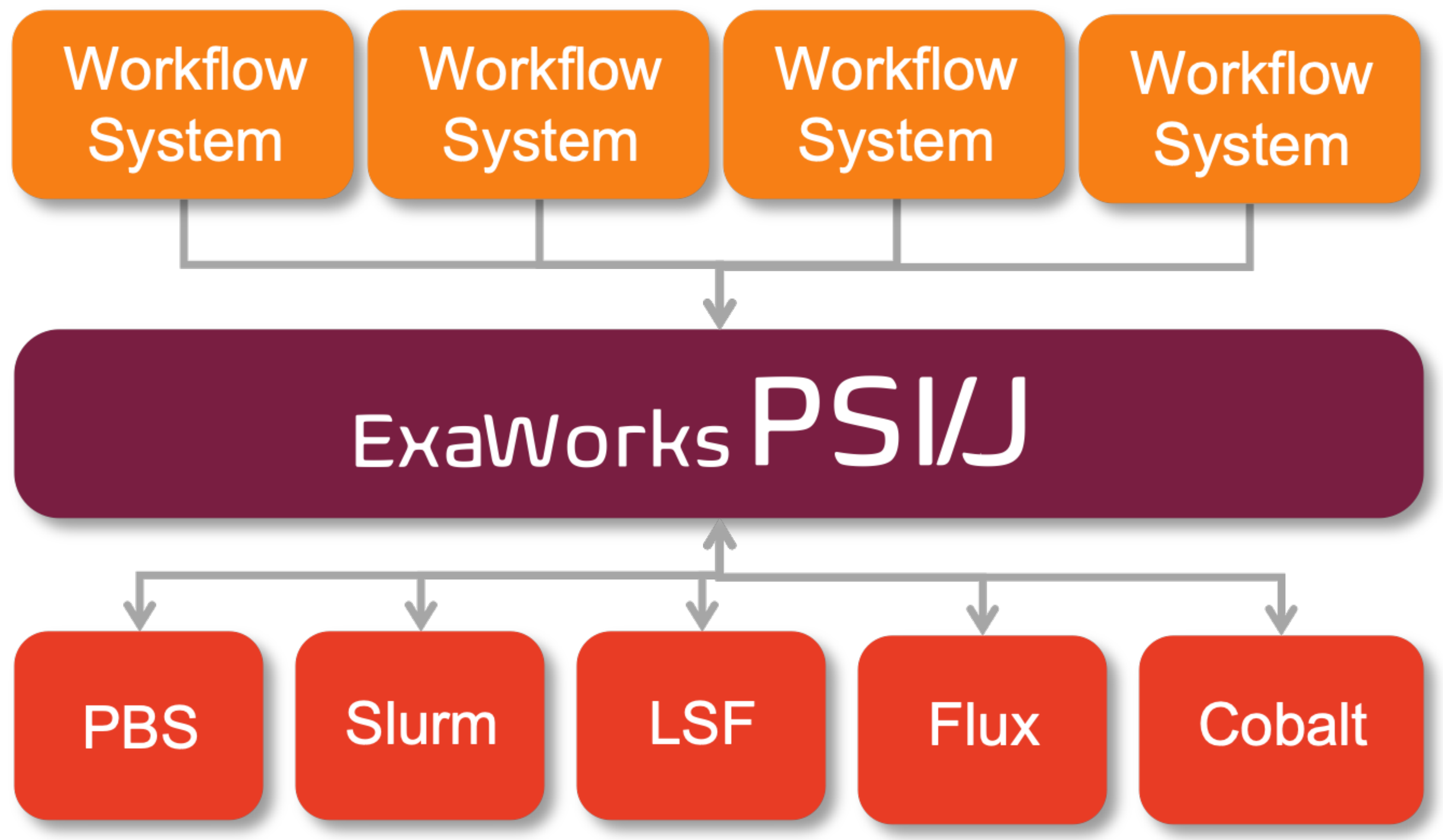
Makefiles use the filesystem for dependency resolution and task tracking…

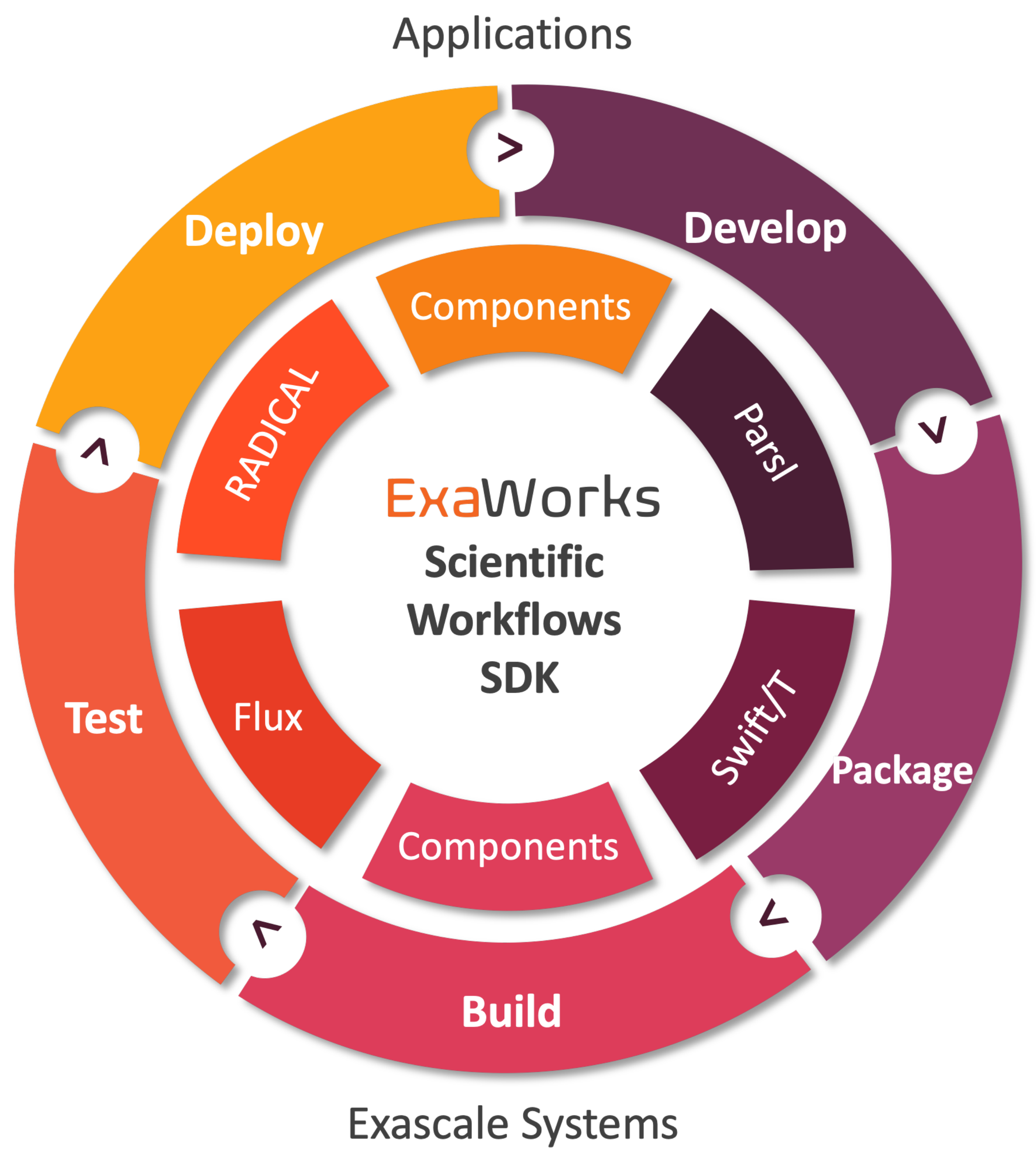
…which is awesome and in the spirit of the UNIX philosophy, until it breaks down at scale.

Alternatives

Exaworks

Next generation technologies for composable and scalable HPC workflows





https://exaworks.org/
https://exaworkssdk.readthedocs.io/

THANK YOU

Please reach out to **rcac-help@purdue.edu** for questions.

See **rcac.purdue.edu/training/workflow_automation** for slides.
See **rcac.purdue.edu/training** for additional topics.
See **rcac.purdue.edu/knowledge** for user-guides.

PURDUE UNIVERSITY | Rosen Center for Advanced Computing