Workflow tradeoffs in the context of cancer phylogeny

Alejandro Duque and Abdullah Syed

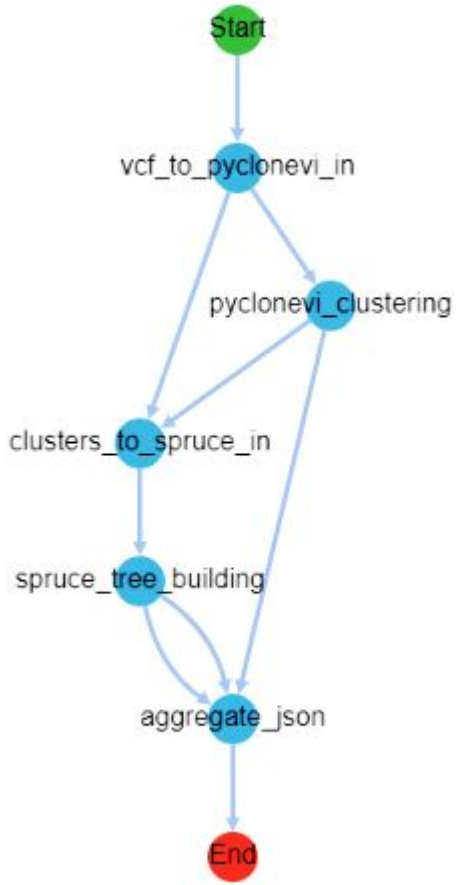
Mentors
Dr. Daniel Katz
Matthew Berry

Collaborators
Dr. Volodymyr Kindratenko
Kastan Day

ILLINOIS
NCSA | National Center for Supercomputing Applications

Context

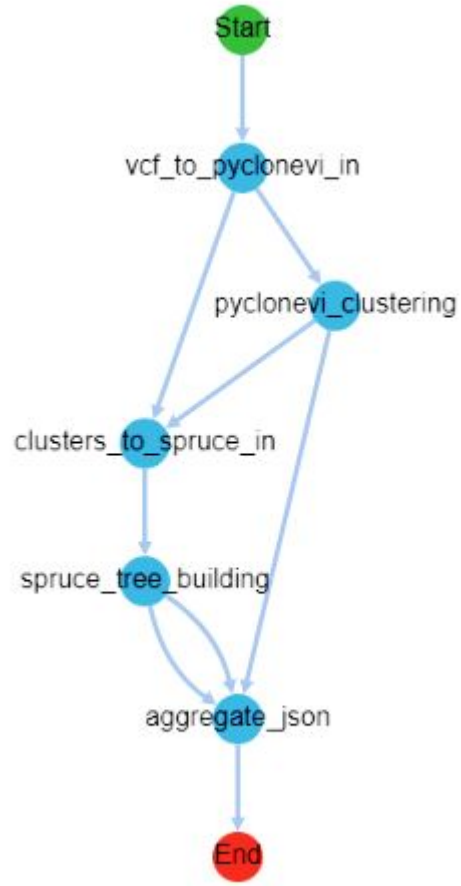
- Cancer phylogenies are graphs that represent the evolutionary relationships and growth of tumors.
- Phylogenetic workflows are pipelines used to build phylogenetic graphs by processing genomic and mutagenic data in a multistep process
- These often use WDL (Workflow Description Language), a bioinformatic framework for executing scientific workflows. The workflow we primarily researched, phyloflow, made heavy usage of WDL.
- Implemented solutions through Parsl, a Python scientific computing framework that enables simplification of workflows, easy parallelization, extension of workflows, and more portability.
- Langchain is a framework that connects LLMs to applications. We have been using it to connect to OpenAI's API.



More on Phyloflow

Workflow steps

- Load a VCF file generated by 'mutect' and its annotated version from VEP (Variant Effect Pipeline).
- Convert the mutations from the VCF file into the required input format for 'pyclone-vi'.
- Execute 'pyclone-vi' to cluster the mutations.
- Adapt the output of the pyclone clustering to be compatible with 'spruce' tree inference.
- Gather the relevant output files and merge them into a JSON file that works with the PhyloDiver visualization tool.



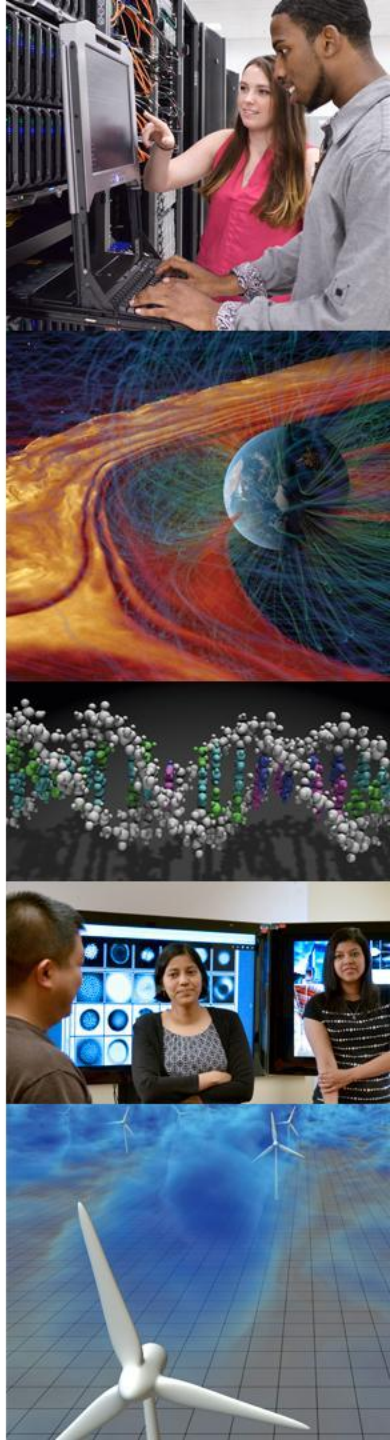
WDL to Parsl Workflow Translation

Why translate the Phyloflow workflow?

- To compare the ease of use of Parsl against WDL on a scientific workflow.
- To test how easy it is to extend a workflow with the parallelizable capabilities of Parsl.
- To make it easier to integrate AI tools developed for Python.

{wdl}

Parsl



ILLINOIS NCSA

WDL to Parsl Workflow Translation

Workflow Translation Process

Understanding Phyloflow workflow

- Tasks and dependencies.

Running the workflow locally

- Setting up running environments.

Translating the workflow to Parsl

- WDL tasks into Parsl Apps.
- Bash scripts into python functions.
- Filesystem Management.
- Containerization.

Documenting the Process:

- https://github.com/grimloc-aduque/Phyloflow-Parsl-Implementation

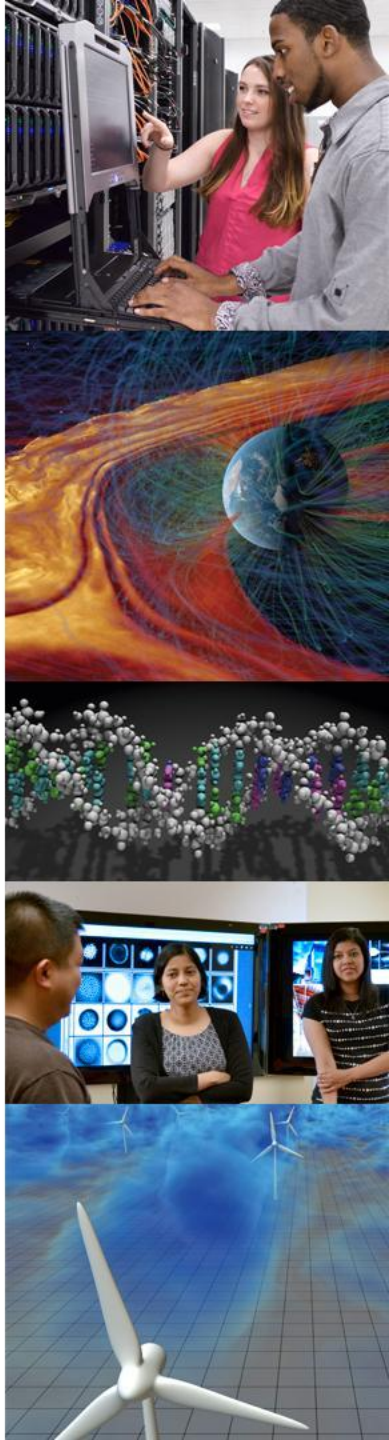
CONDA

docker

@python_app

@bash_app

ILLINOIS NCSA

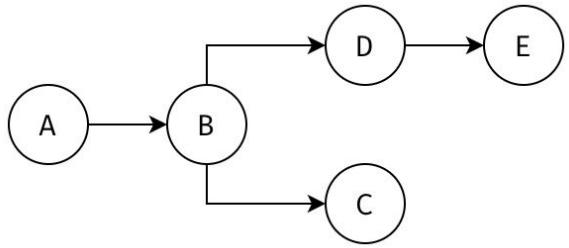


WDL to Parsl Workflow Translation

Developer Experience with Parsl

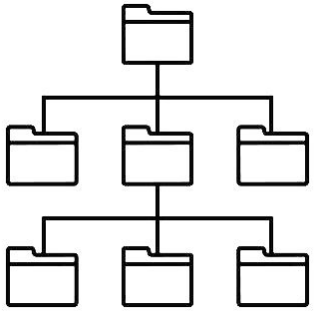
Parsl Advantages

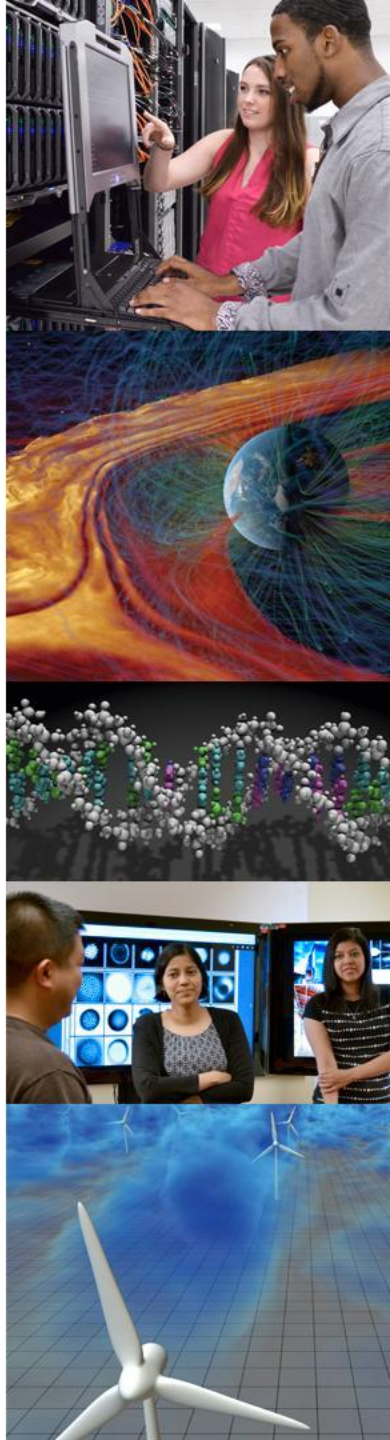
- Finer control over the file dependencies.
- Easy to extend with native python functionality.
- Straightforward parallelization.



Parsl Disadvantages

- Requires more experience with the filesystem.
- Harder to run tasks with conflicting environment dependencies.
- Retrieving inputs and outputs by indexing an array is confusing and error prone.





ILLINOIS NCSA

Workflow Integration with AI

Integration of AI in phylogenetic workflows

Explore ways to use Langchain along with OpenAI LLM's

Workflow translation

- Generate Parsl code from existing WDL workflows.

Workflow generation

- Generate Parsl code with NLP.
- Zero-shot and One-shot prompting.
- RepoReader for document indexing.

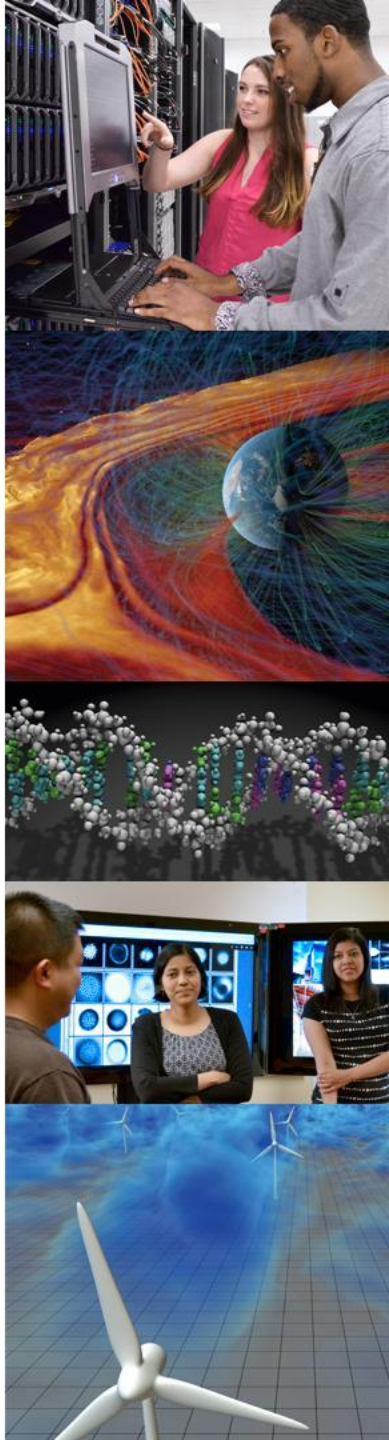
Workflow usage

- Execute workflow tasks using NLP.
- OpenAI function calling.

LangChain

OpenAI

ILLINOIS NCSA

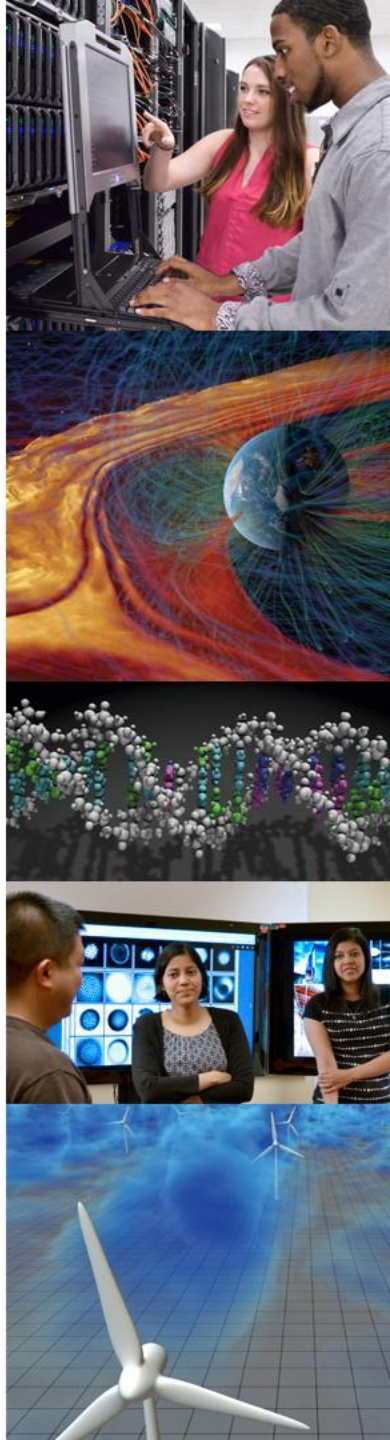


Workflow Integration with AI

Next Steps

- Concatenate function call requests to compose multiple tasks.
- Explore ways to use Langchain along with function calling.
- Implement the Variant Effect Predictor as a node of the workflow.

ILLINOIS NCSA



References

- Phyloflow: https://github.com/ncsa/phyloflow
- Parsl: https://parsl.readthedocs.io/en/stable/
- Langchain: https://python.langchain.com/docs/
- OpenAI API: https://platform.openai.com/docs/api-reference
- Function calling: https://openai.com/blog/function-calling-and-other-api-updates