Duct Tape, DAGs, and Determination:

Snakemake at the Edge of HTCondor

or

"I Brought a Snake to a Bird Fight"

Justin Hiemstra

CHTC HTCondor Software Suite PATh

The two types of "Research Software"

Type 1

Type 2

The two types of "Research Software"

Type 1

Type 2

Justin Hiemstra

Research Software Engineer

Morgridge Institute For Research

HTCondor Software Suite

Pelican Platform

The two types of "Research Software"

Type 1

Justin Hiemstra

Research Software Engineer

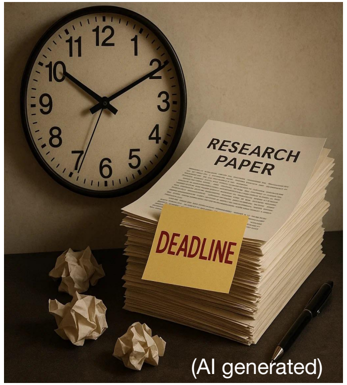
Morgridge Institute For Research

HTCondor
Software Suite

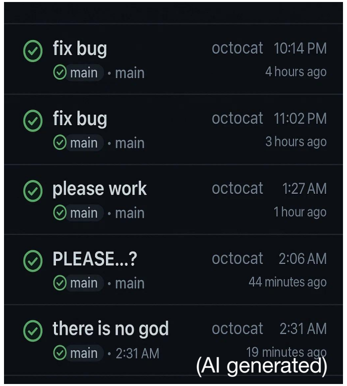
Pelican
Platform

Type 2

(AI generated)



(AI generated)



```
✓ fix bug                octocat  10:14 PM
  ✓ main · main                 4 hours ago
✓ fix bug                octocat  11:02 PM
  ✓ main · main                 3 hours ago
✓ please work            octocat   1:27 AM
  ✓ main · main                  1 hour ago
✓ PLEASE...?             octocat   2:06 AM
  ✓ main · main              44 minutes ago
✓ there is no god        octocat   2:31 AM
  ✓ main · 2:31 AM           19 minutes ago
```

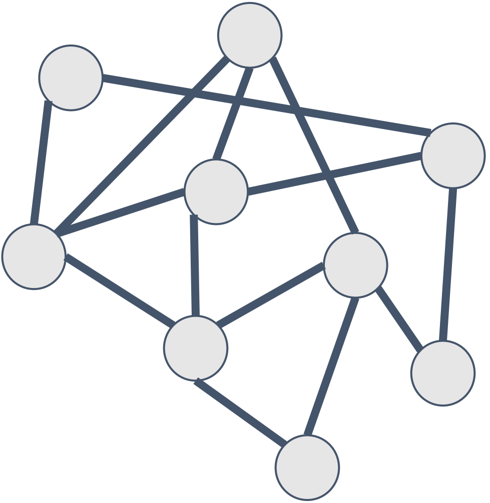
(AI generated)

This is a story of running *other* people's "type 2" research software at scale

SPRAS – The "Signaling Pathway Reconstruction Streamliner"

There is…

SPRAS – The "Signaling Pathway Reconstruction Streamliner"

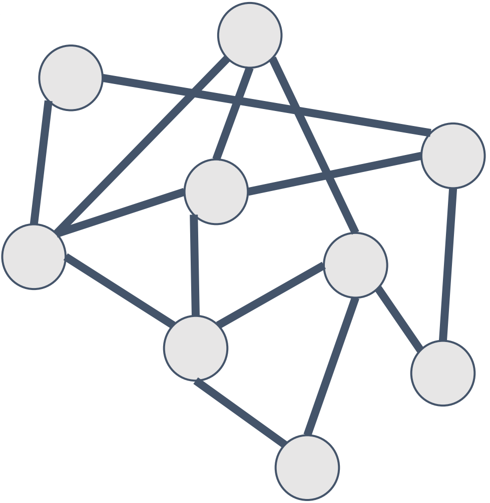


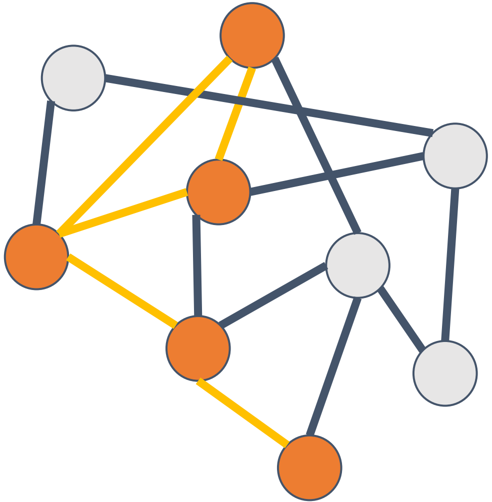
There is a lot of experimental protein-protein interaction data

SPRAS – The "Signaling Pathway Reconstruction Streamliner"

There is a lot of experimental protein-protein interaction data

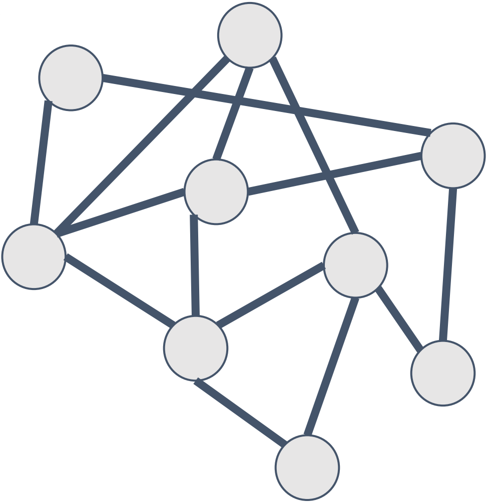
It is often difficult to know which path from one node to another is the most likely "signaling pathway"





CHTC HTCondor Software Suite PATh

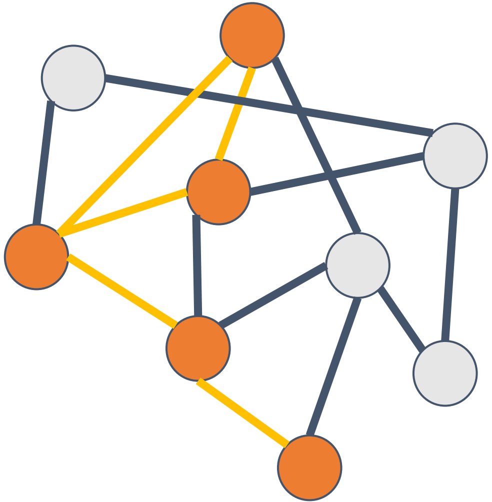
SPRAS – The "Signaling Pathway Reconstruction Streamliner"



There is a lot of experimental protein-protein interaction data

It is often difficult to know which path from one node to another is the most likely "signaling pathway"

There are many published "type 2" algorithms that try to answer this question, but few are widely adopted



CHTC HTCondor Software Suite PATh

SPRAS – The "Signaling Pathway Reconstruction Streamliner"

What if we created a tool that streamlined running as many of these algorithms as possible?

CHTC HTCondor Software Suite PATh

SPRAS – The "Signaling Pathway Reconstruction Streamliner"

What if we created a tool that streamlined running as many of these algorithms as possible?

"*All problems in computer science can be solved by another level of indirection, except for the problem of too many layers of indirection.*" – David Wheeler

CHTC HTCondor Software Suite PATh

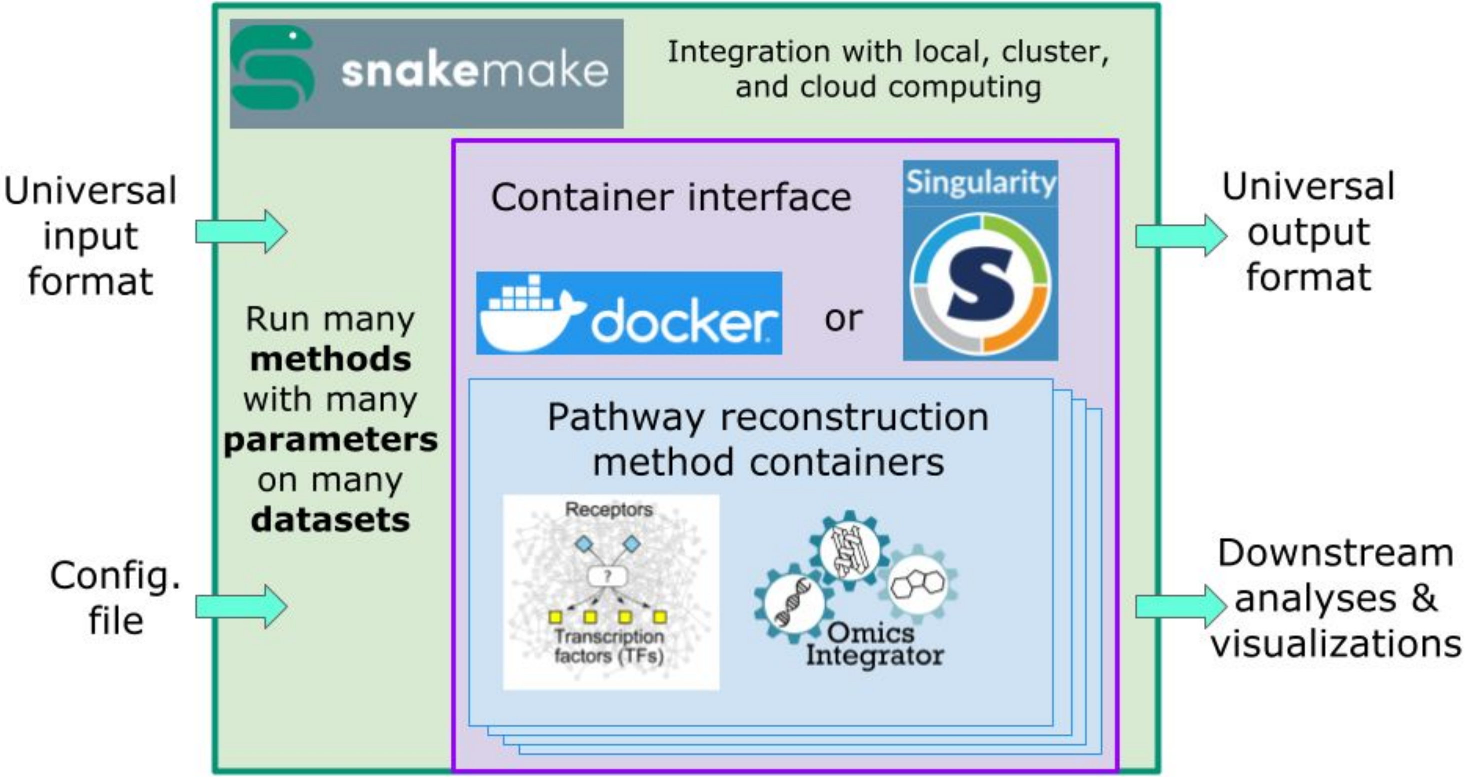
SPRAS – The "Signaling Pathway Reconstruction Streamliner"

What if we created a tool that streamlined running as many of these algorithms as possible?

"By indirections find directions out" - Polonius, *Hamlet*

CHTC HTCondor Software Suite PATh

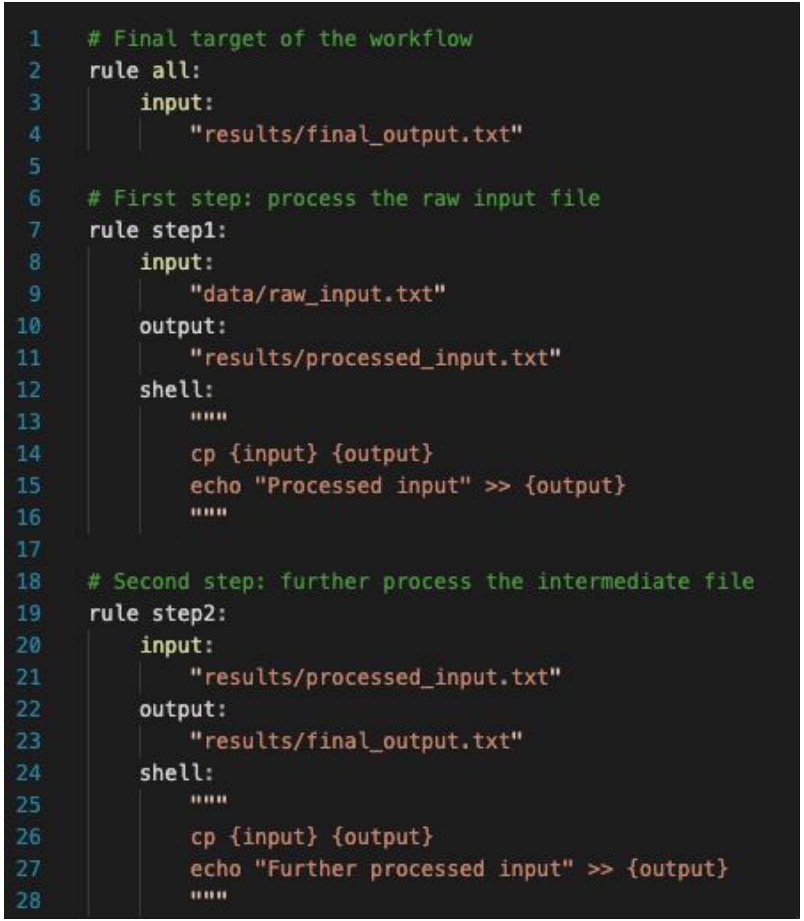
Finding Directions Out


snakemake
Integration with local, cluster, and cloud computing
Universal input format
Config. file
Run many **methods** with many **parameters** on many **datasets**
Container interface
docker
or
Singularity
Pathway reconstruction method containers
Receptors
?
Transcription factors (TFs)
Omics Integrator
Universal output format
Downstream analyses & visualizations

Workflow Management

snakemake

- Rule-based & declarative
- Dependency resolution – builds a DAG of work
- Capable of job skipping
- Pythonic
- *Widely used in the Bioinformatics community*

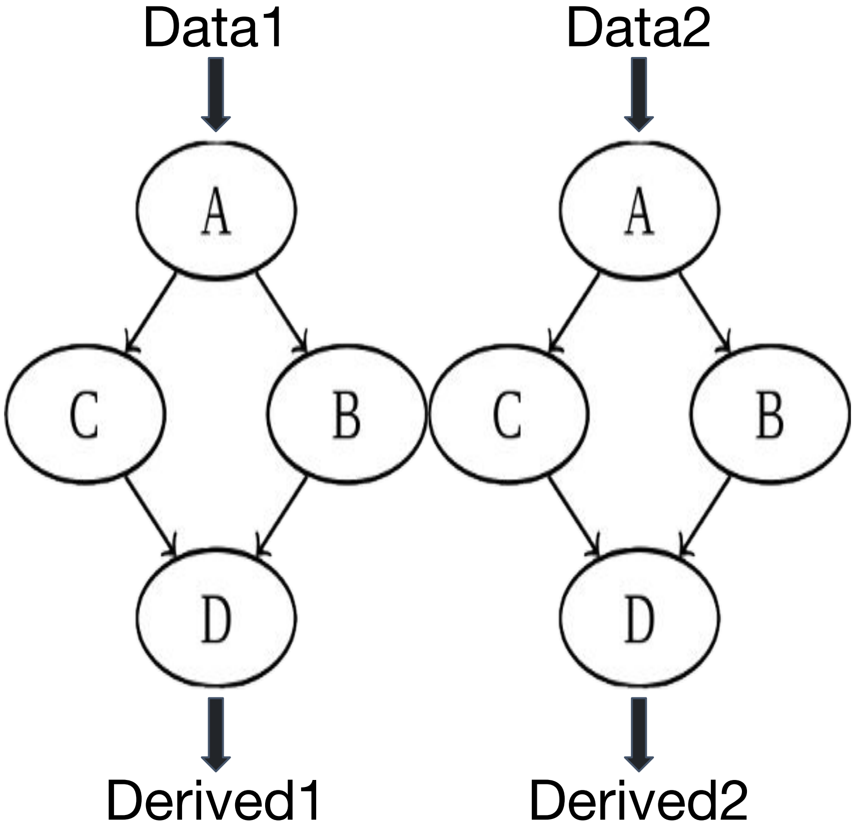


```
# Final target of the workflow
rule all:
    input:
        "results/final_output.txt"

# First step: process the raw input file
rule step1:
    input:
        "data/raw_input.txt"
    output:
        "results/processed_input.txt"
    shell:
        """
        cp {input} {output}
        echo "Processed input" >> {output}
        """

# Second step: further process the intermediate file
rule step2:
    input:
        "results/processed_input.txt"
    output:
        "results/final_output.txt"
    shell:
        """
        cp {input} {output}
        echo "Further processed input" >> {output}
        """
```

Workflow Management

snakemake

- Rule-based & declarative
- Dependency resolution – builds a DAG of work
- Capable of job skipping
- Pythonic
- *Widely used in the Bioinformatics community*



Why not just use DAGMan?

SPRAS existed before any attempt to integrate with HTCondor

An early design constraint was to keep the overall architecture flexible **without** requiring that SPRAS users have access to HTCondor

What would it take…

What would it take…

… to run **one** algorithm with SPRAS on **one** dataset as only **one** job on **one** EP?

What would it take…

… to run **one** algorithm with SPRAS on **one** dataset as only **one** job on **one** EP?

1. Containerize the software
2. Create a submit file
3. Submit
4. Solve problems, squash bugs
5. `goto 3;`

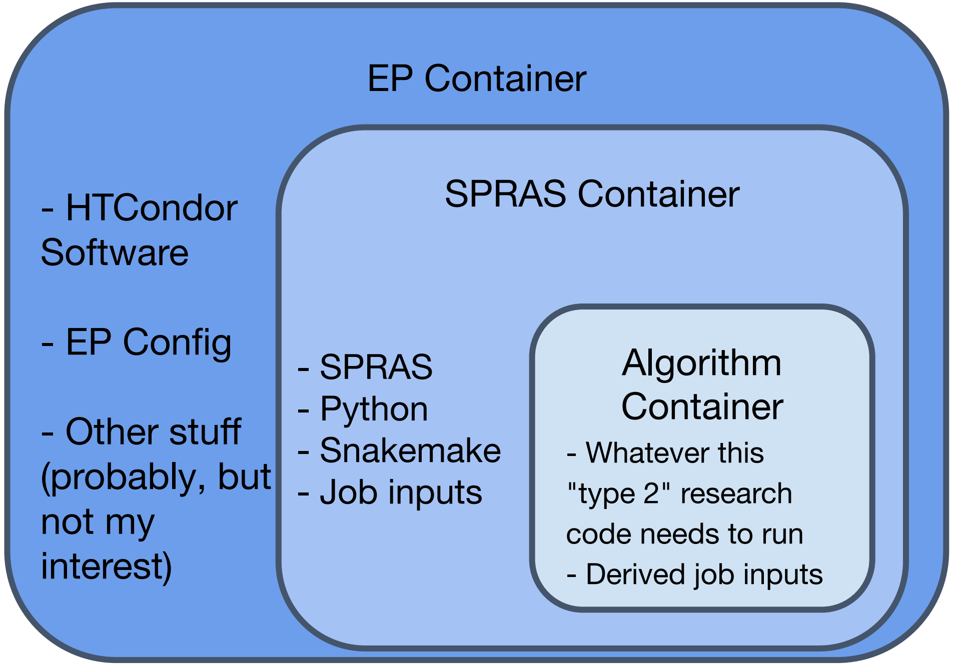
What would it take…

… to run **one** algorithm with SPRAS on **one** dataset as only **one** job on **one** EP?

1. Containerize the software
2. Create a submit file
3. Submit
4. Solve problems, squash bugs
5. `goto 3;`

The first appreciable hurdle was that containerizing SPRAS itself introduced a need to run nested containers

Nested Containers in the OSPool



EP Container

- HTCondor Software
- EP Config
- Other stuff (probably, but not my interest)

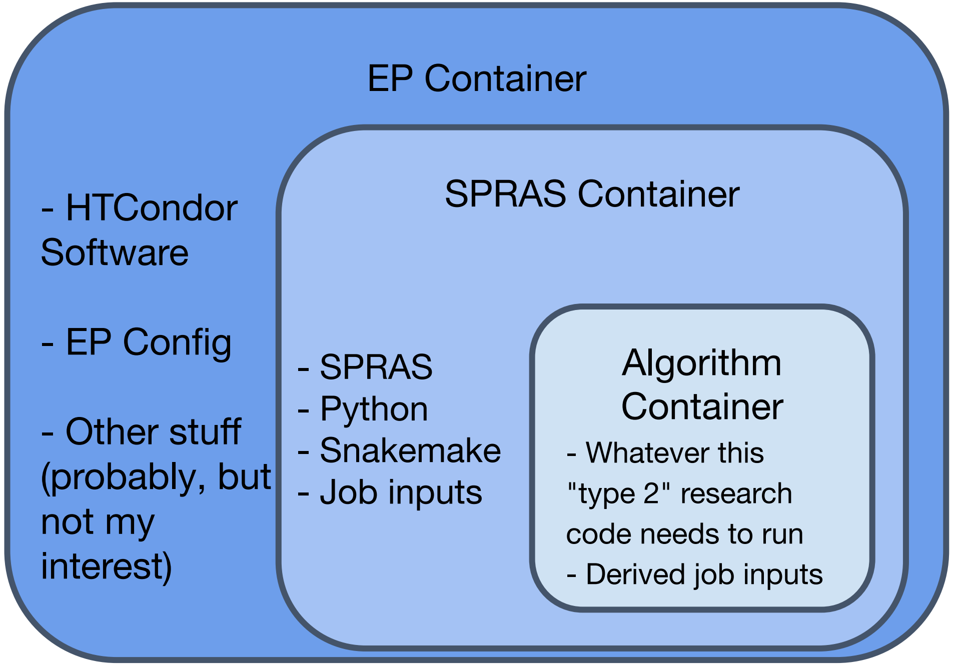
SPRAS Container

- SPRAS
- Python
- Snakemake
- Job inputs

Algorithm Container

- Whatever this "type 2" research code needs to run
- Derived job inputs

Nested Containers in the OSPool



EP Container

- HTCondor Software

- EP Config

- Other stuff (probably, but not my interest)

SPRAS Container

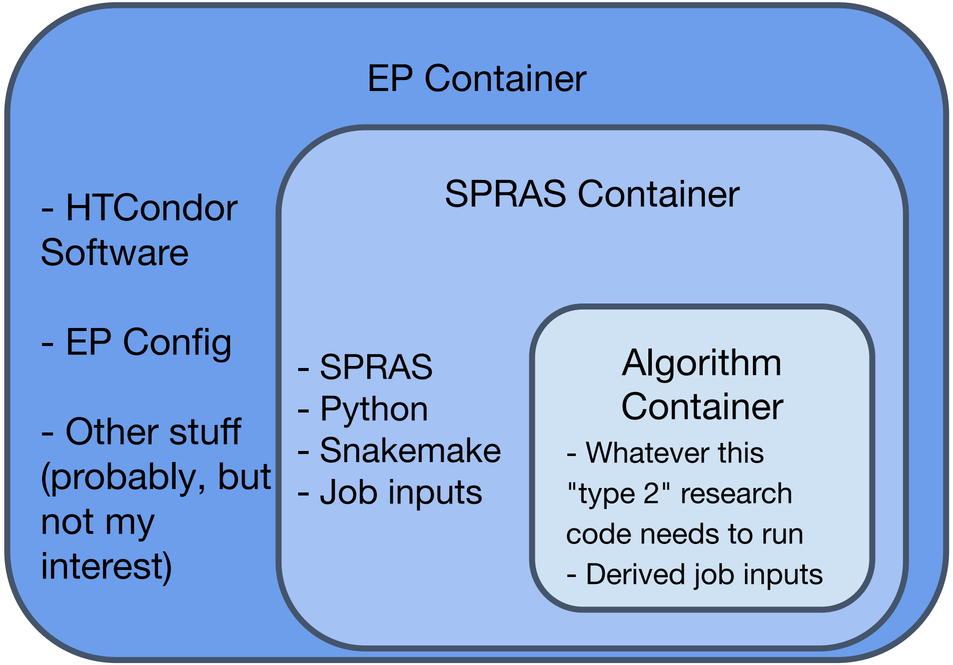
- SPRAS
- Python
- Snakemake
- Job inputs

Algorithm Container

- Whatever this "type 2" research code needs to run
- Derived job inputs

Nested containers in a heterogeneous, unprivileged environment took me a while to figure out

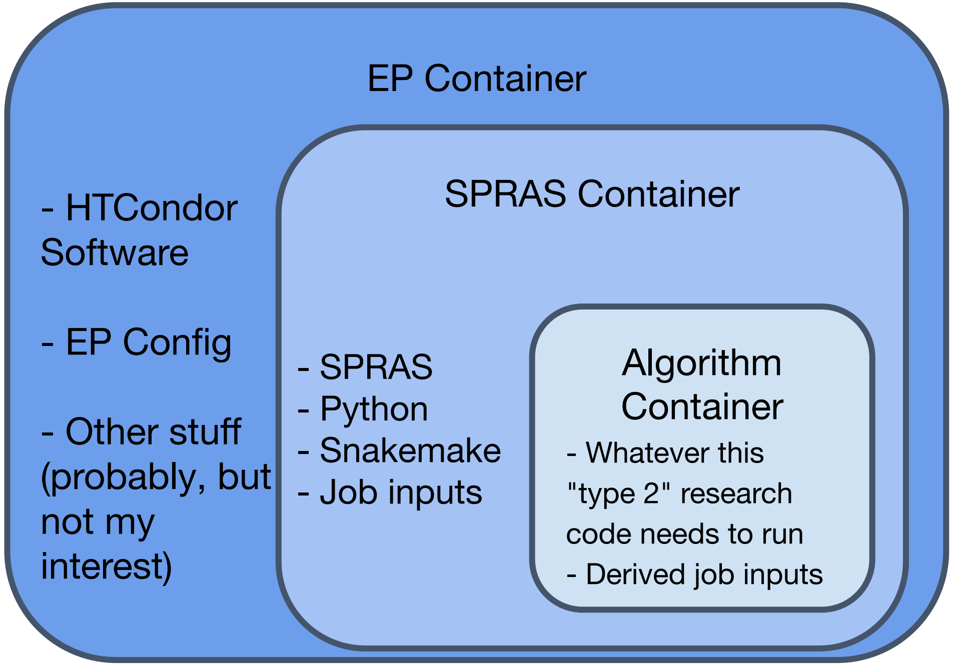
Nested Containers in the OSPool



Nested containers in a heterogeneous, unprivileged environment took me a while to figure out

- Docker is not your friend, but Apptainer is
- Barring significant architectural changes meant converting Docker→ .sif in the job
- A very opaque, difficult environment to develop in/for
- **The secret**: starting apptainer from "unpacked" .sif images (`--sandbox`)

Nested Containers in the OSPool



Nested containers in a heterogeneous, unprivileged environment took me a while to figure out

- Docker is not your friend, but Apptainer is
- Barring significant architectural changes meant converting Docker→ .sif in the job
- A very opaque, difficult environment to develop in/for
- **The secret**: starting apptainer from "unpacked" .sif images (`--sandbox`)

Everything is simple when you know how

Teaching the Snake to Fly The Bird

The next step was getting Snakemake to manage its own "jobs"

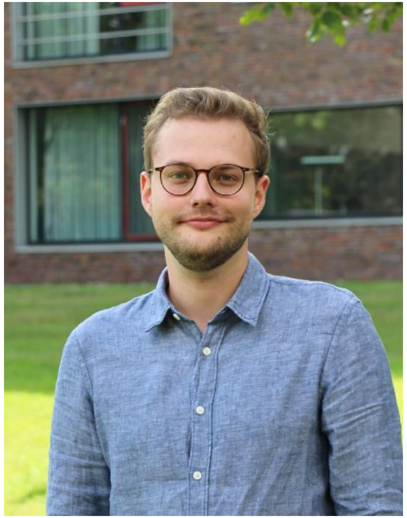
Luckily, right around this time Snakemake made an "Executor" plugin interface

To my surprise, I wasn't the only one working on this!

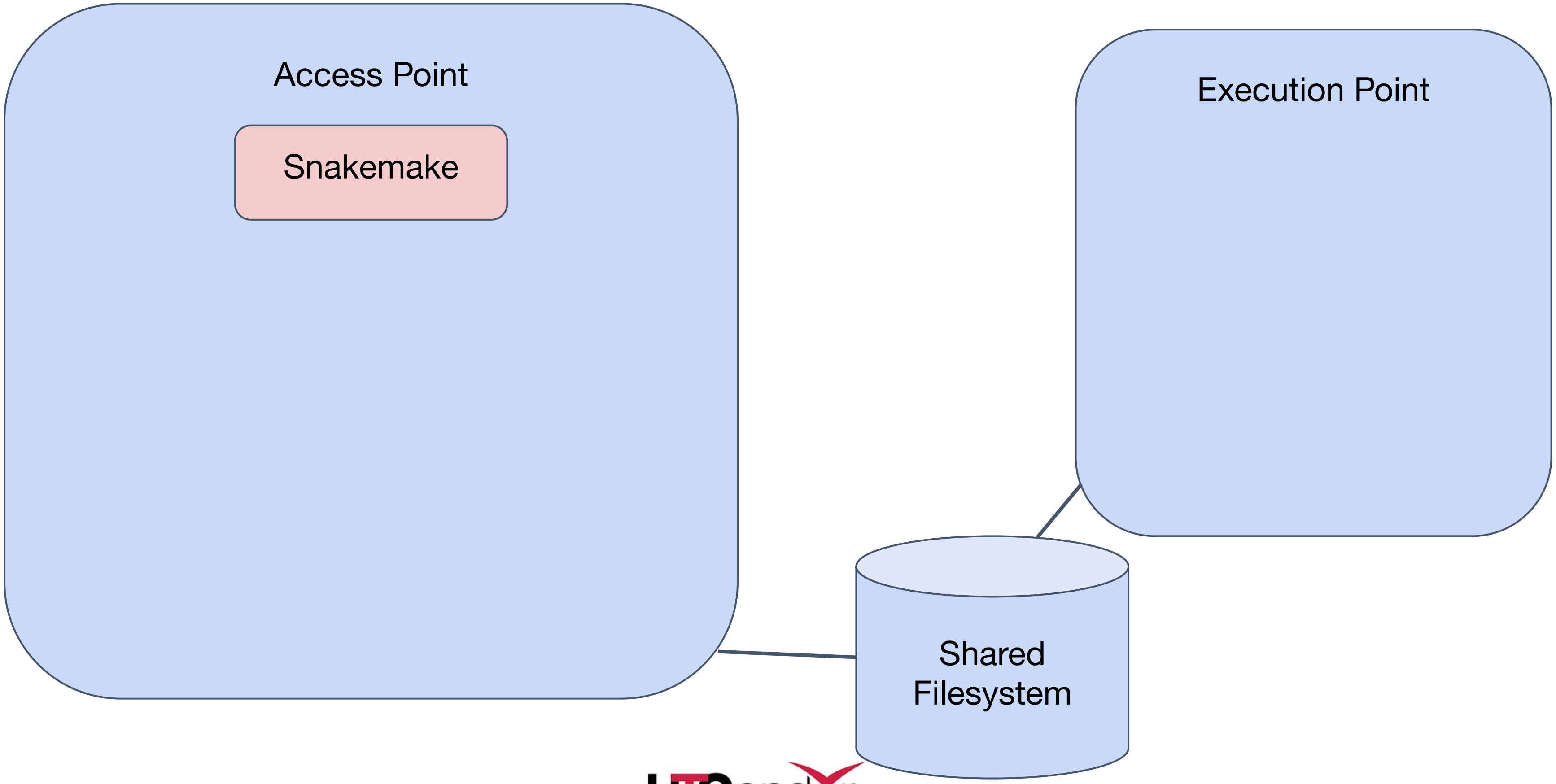
This lets Snakemake submit each of its "jobs" as an HTCondor job using the Python bindings

The Executor manages/monitors jobs and output so it knows when to submit the next unit of work

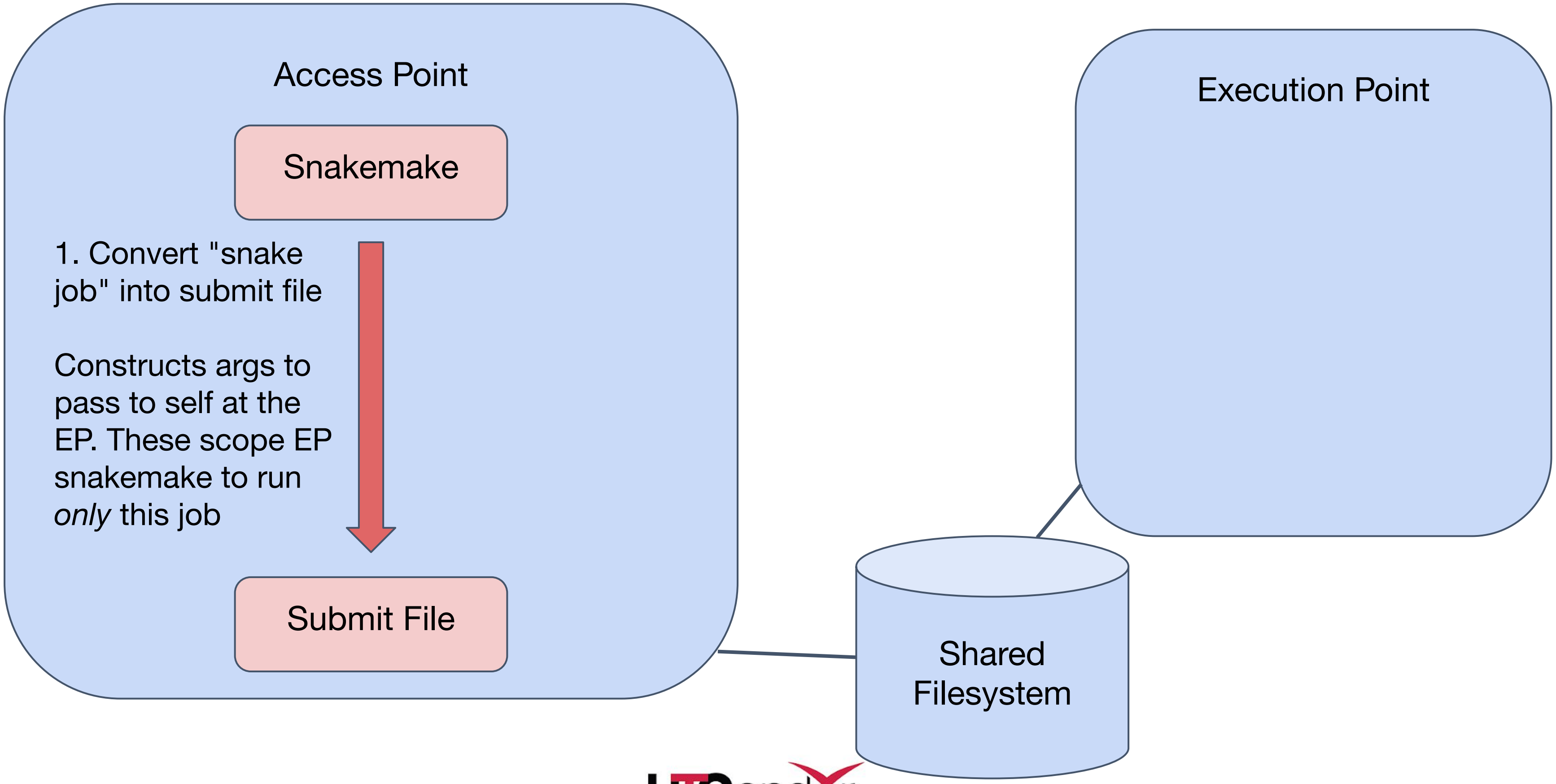
Jannis Speer @ University of Dortmund



Original HTCondor Executor Author



CHTC HTCondor Software Suite PATh



Access Point

Snakemake

1. Convert "snake job" into submit file

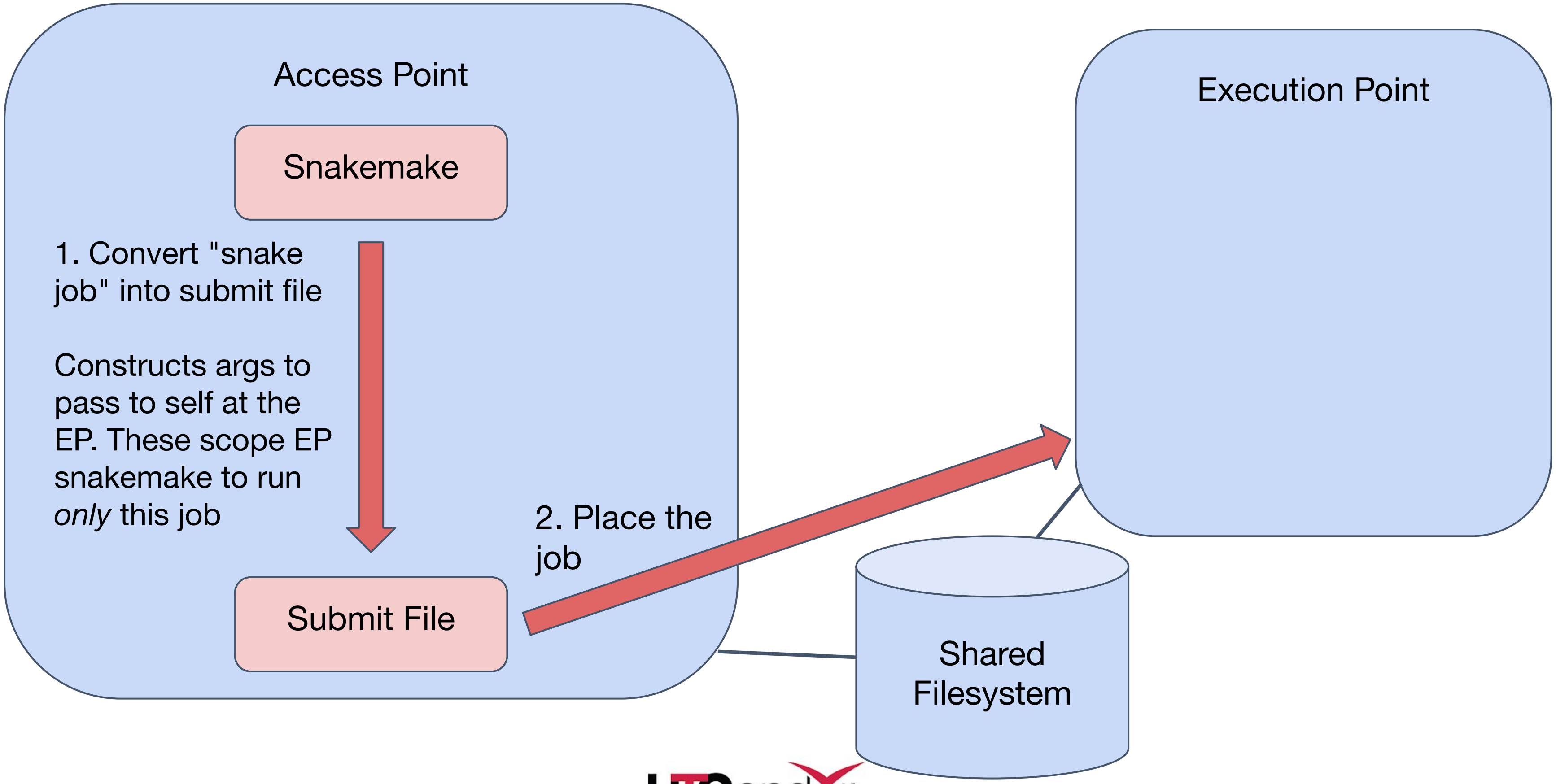
Constructs args to pass to self at the EP. These scope EP snakemake to run *only* this job

Submit File

Execution Point

Shared Filesystem

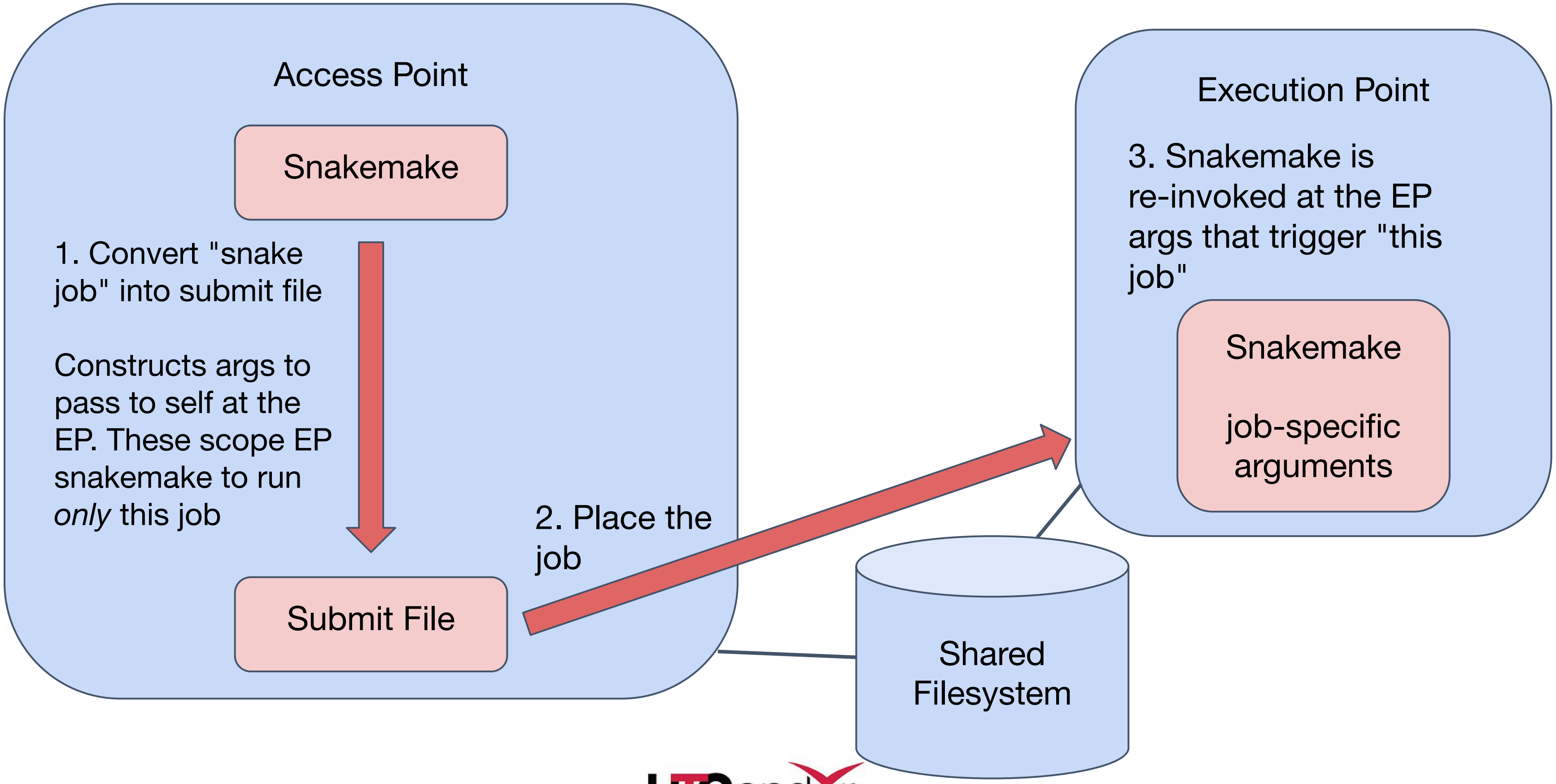
CHTC HTCondor Software Suite PATh



1. Convert "snake job" into submit file

Constructs args to pass to self at the EP. These scope EP snakemake to run *only* this job

2. Place the job



Access Point

Snakemake

1. Convert "snake job" into submit file

Constructs args to pass to self at the EP. These scope EP snakemake to run *only* this job

Submit File

2. Place the job

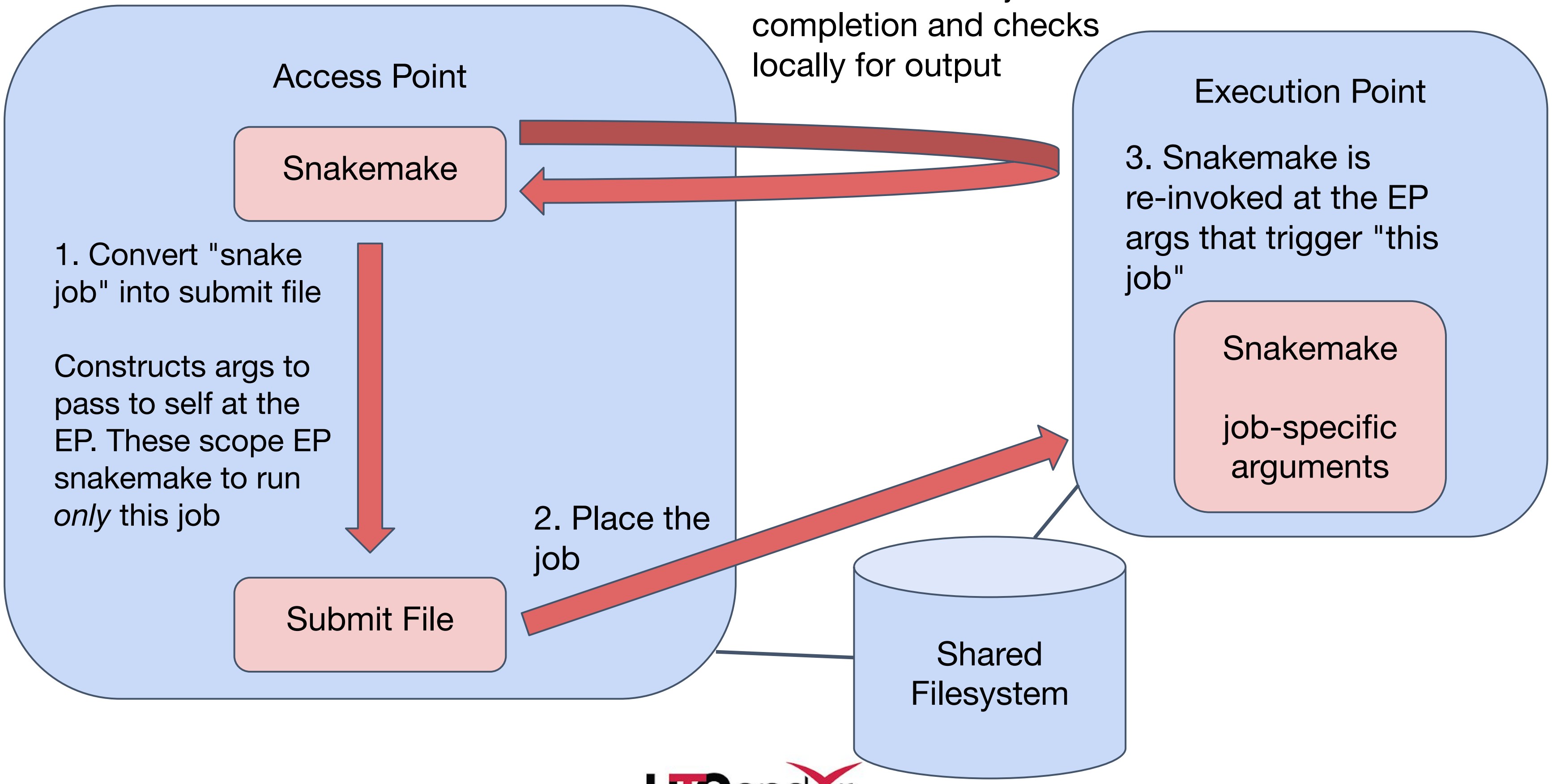
Execution Point

3. Snakemake is re-invoked at the EP args that trigger "this job"

Snakemake

job-specific arguments

Shared Filesystem



Access Point

Snakemake

1. Convert "snake job" into submit file

Constructs args to pass to self at the EP. These scope EP snakemake to run *only* this job

Submit File

2. Place the job

Execution Point

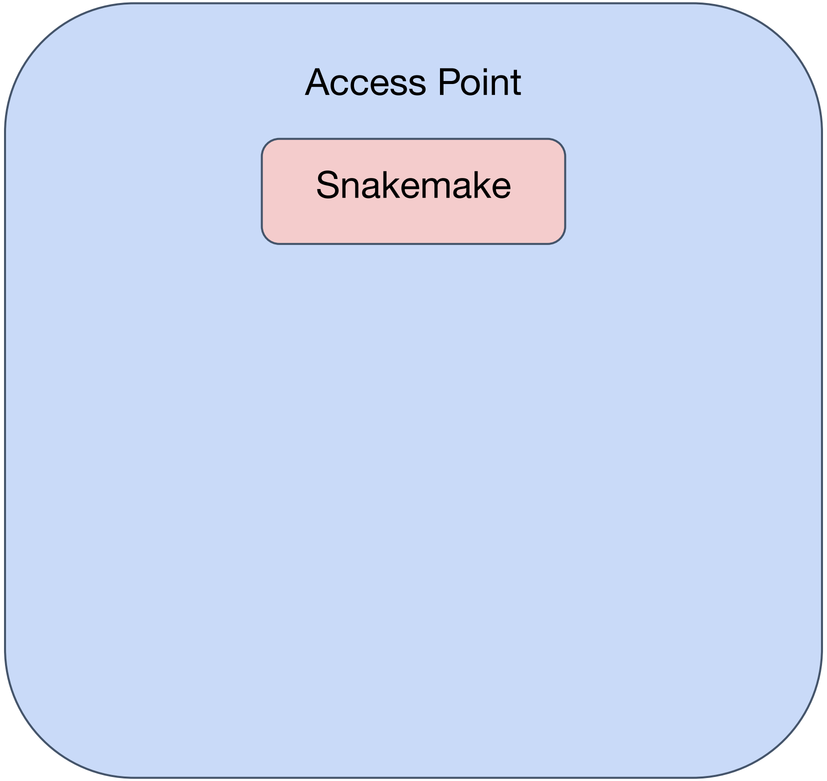
3. Snakemake is re-invoked at the EP args that trigger "this job"

Snakemake

job-specific arguments

4. Snakemake at the AP monitors remote job for completion and checks locally for output

Shared Filesystem



5. When output is complete, move onto next job using previous output as new input

Shared Filesystems, Shared Suffering

Shared Filesystems, Shared Suffering

When you *assume* a shared filesystem, you make an @$$ out of you, me and the rest of the cluster

Shared Filesystems, Shared Suffering

When you *assume* a shared filesystem, you make an @$$ out of you, me and the rest of the cluster

It looked like Snakemake already had had a runtime flag for me

```
--shared-fs-usage none
```

Would it work for me off the shelf??

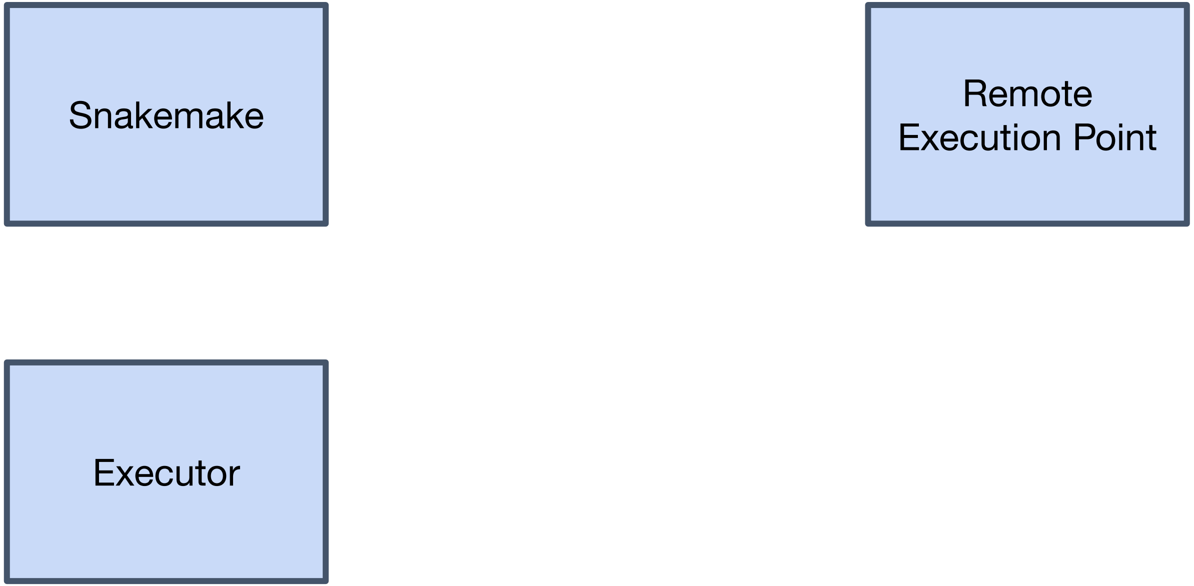
no

What exactly *does* this option do?

Some of the "job-specific" args are modified to switch absolute paths to relative → A promising thing to see!

It does *try* to handle transferring input/output and the executable… just in the wrong way

Snakemake Transfer Model

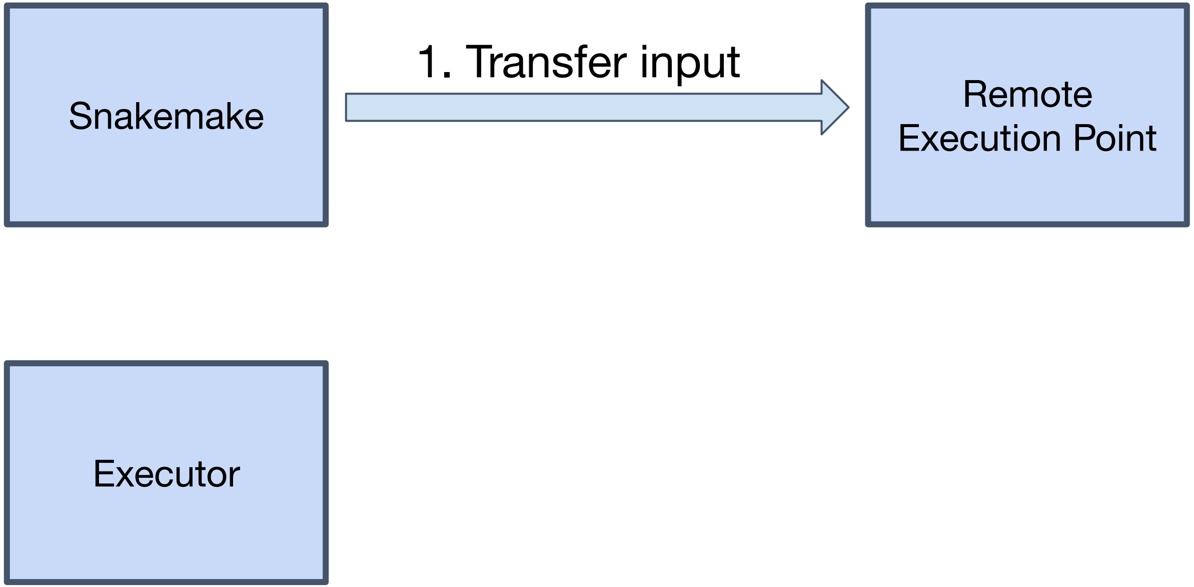


Snakemake

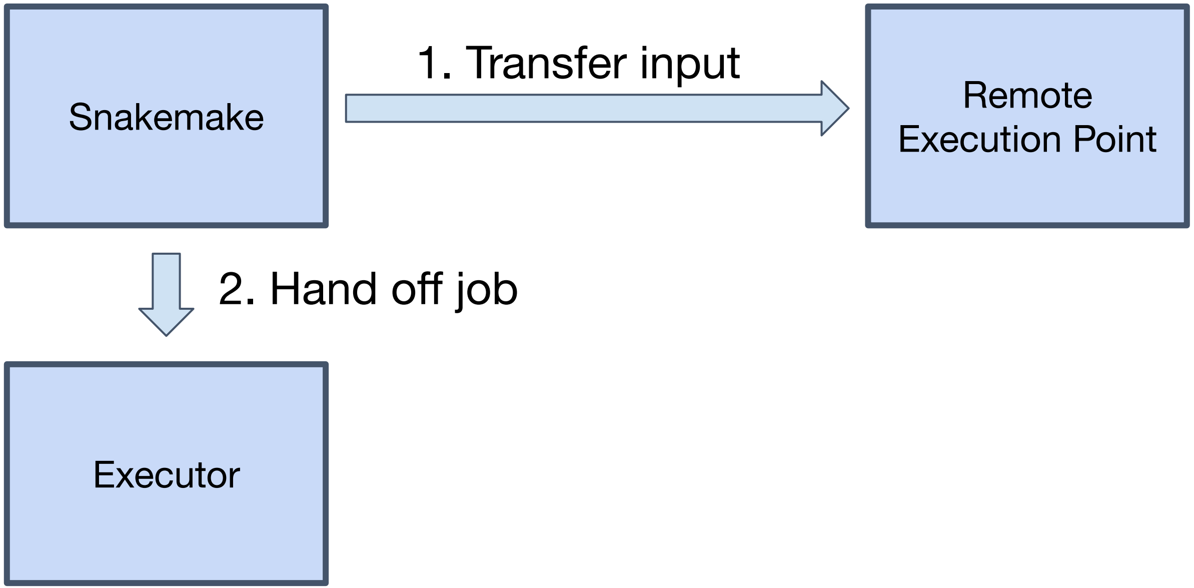
Remote Execution Point

Executor

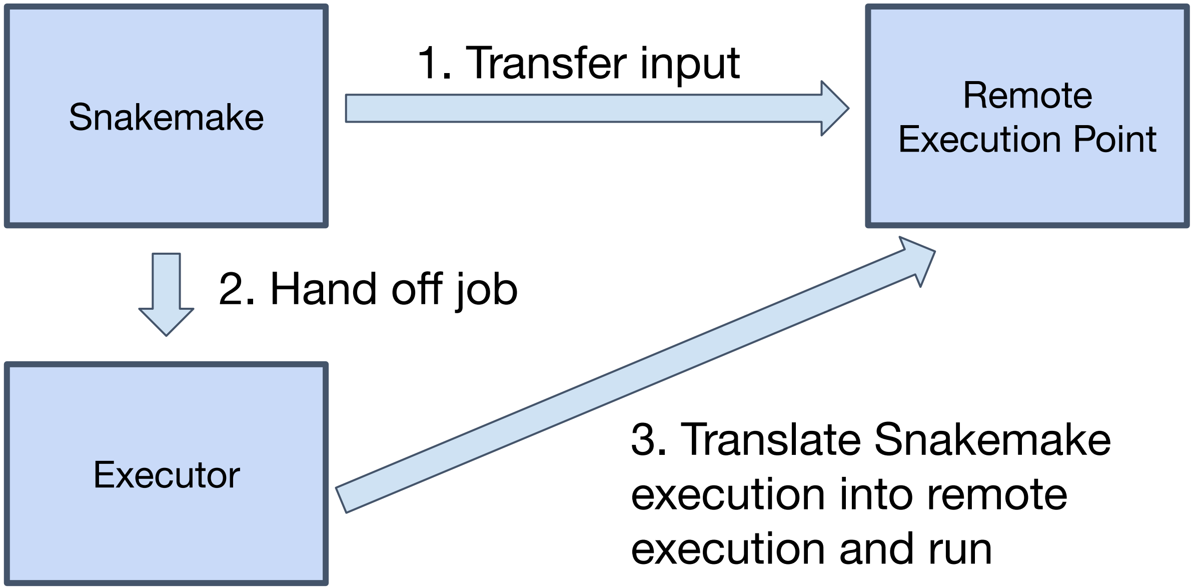
Snakemake Transfer Model



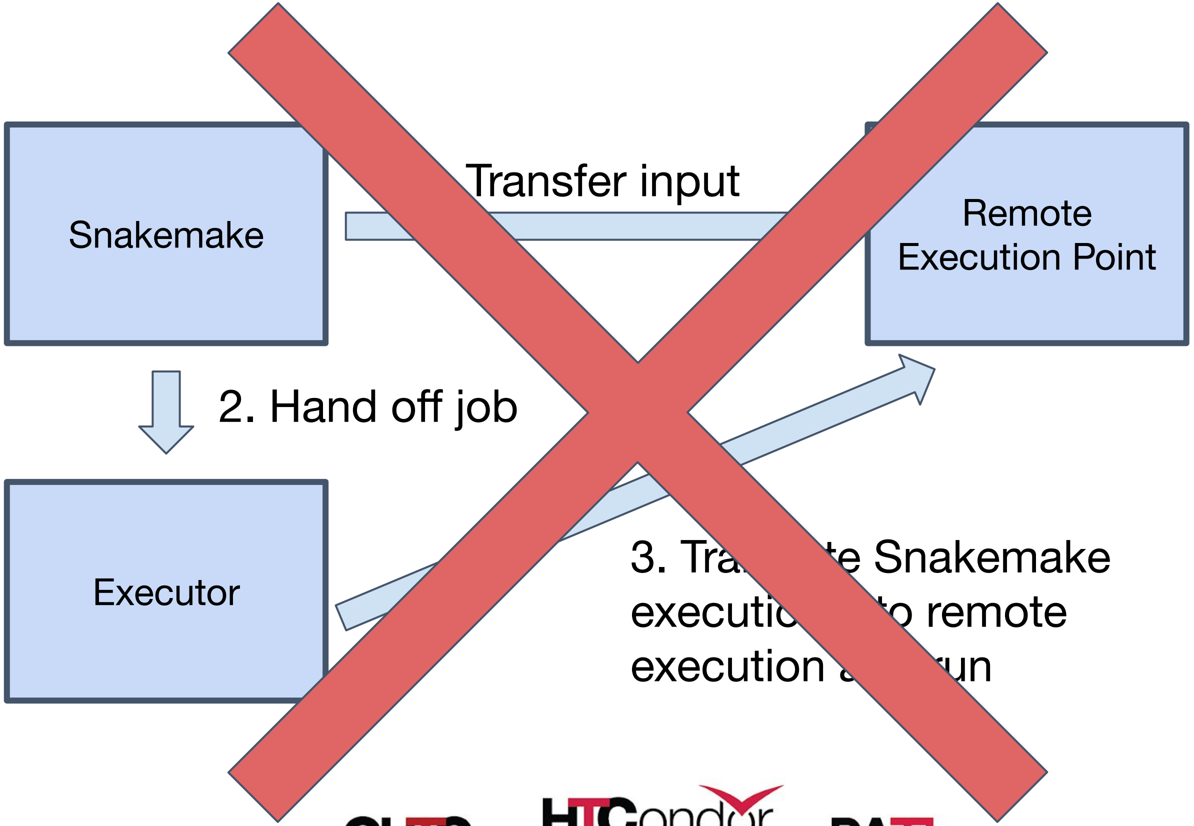
Snakemake Transfer Model



Snakemake Transfer Model

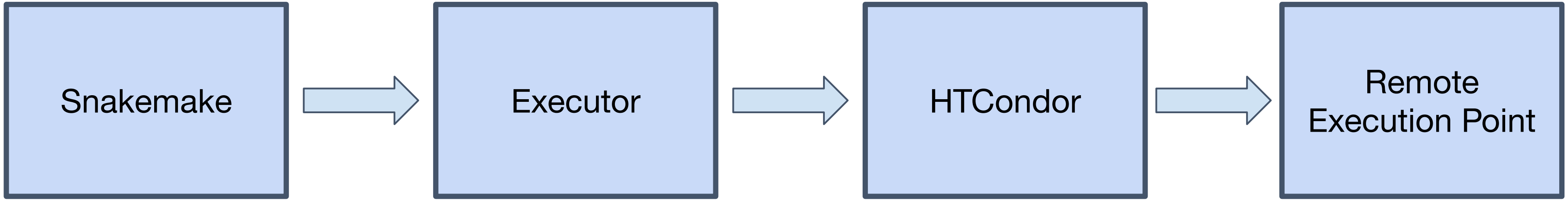


Snakemake Transfer Model



Adjusted Snakemake Transfer Model*

(*required upstream changes – Snakemake maintainers are *really* great to work with!)



1. Hand off job, telling the executor plugin the executor knows how to transfer its own input/output

2. The executor adds these to `transfer_input_files`, `transfer_output_files`

3. HTCondor does its thing and everyone's happy

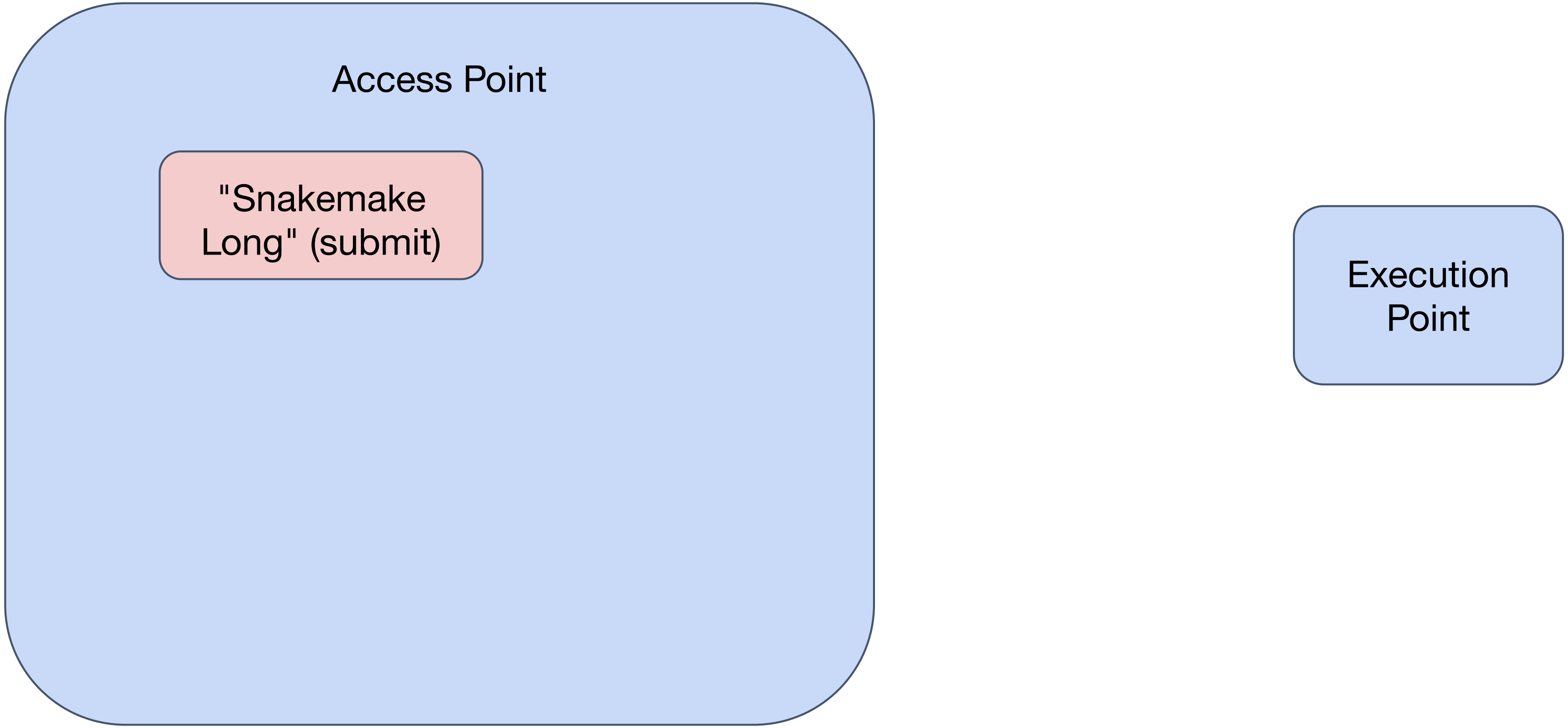
Liftoff

After working through all these (more nuanced than I'm presenting) issues over the course of a few months… things started working

But there were still two things that annoyed me about the setup

1. Running `condor_watch_q` doesn't pick up jobs submitted after the initial invocation
2. If I shut my laptop…

"Snakemake Long"

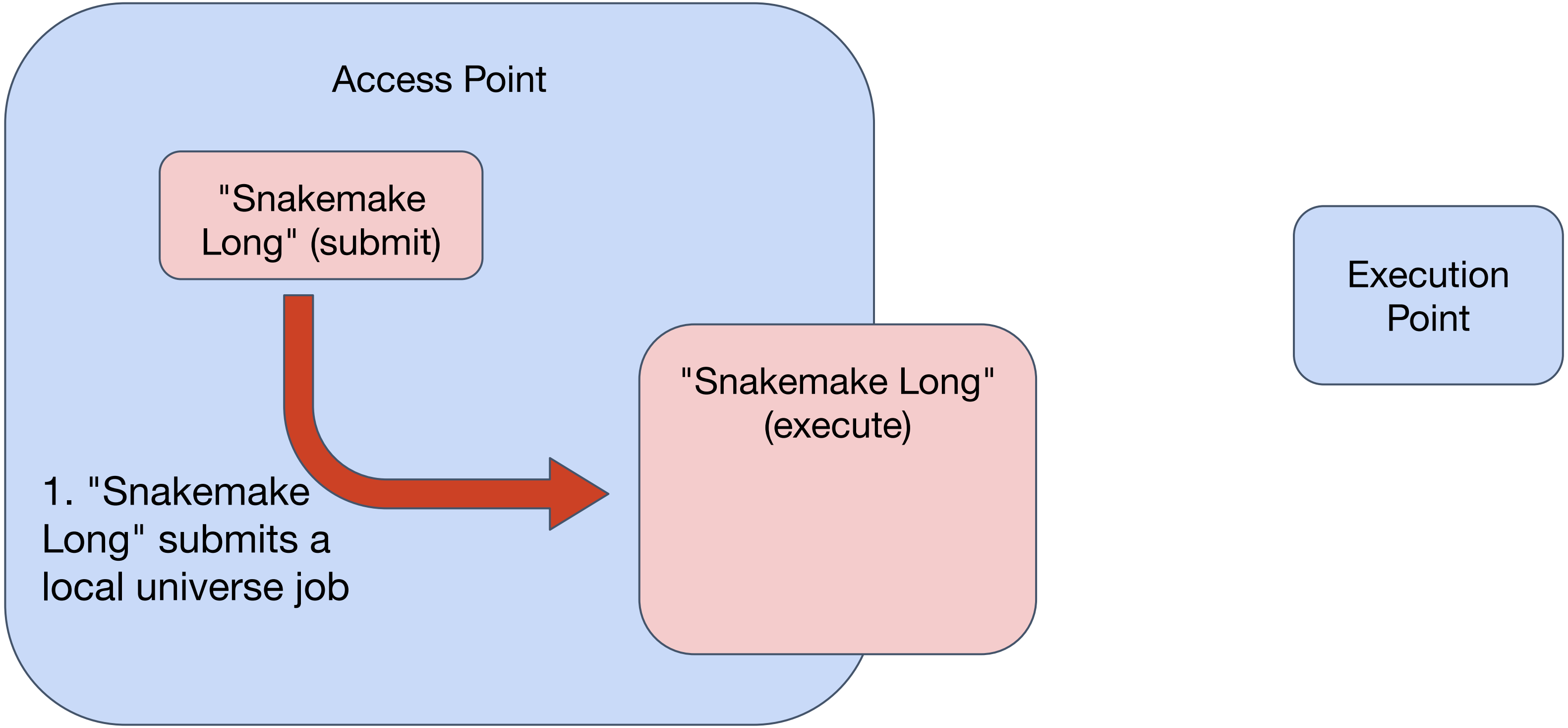


Access Point

"Snakemake Long" (submit)

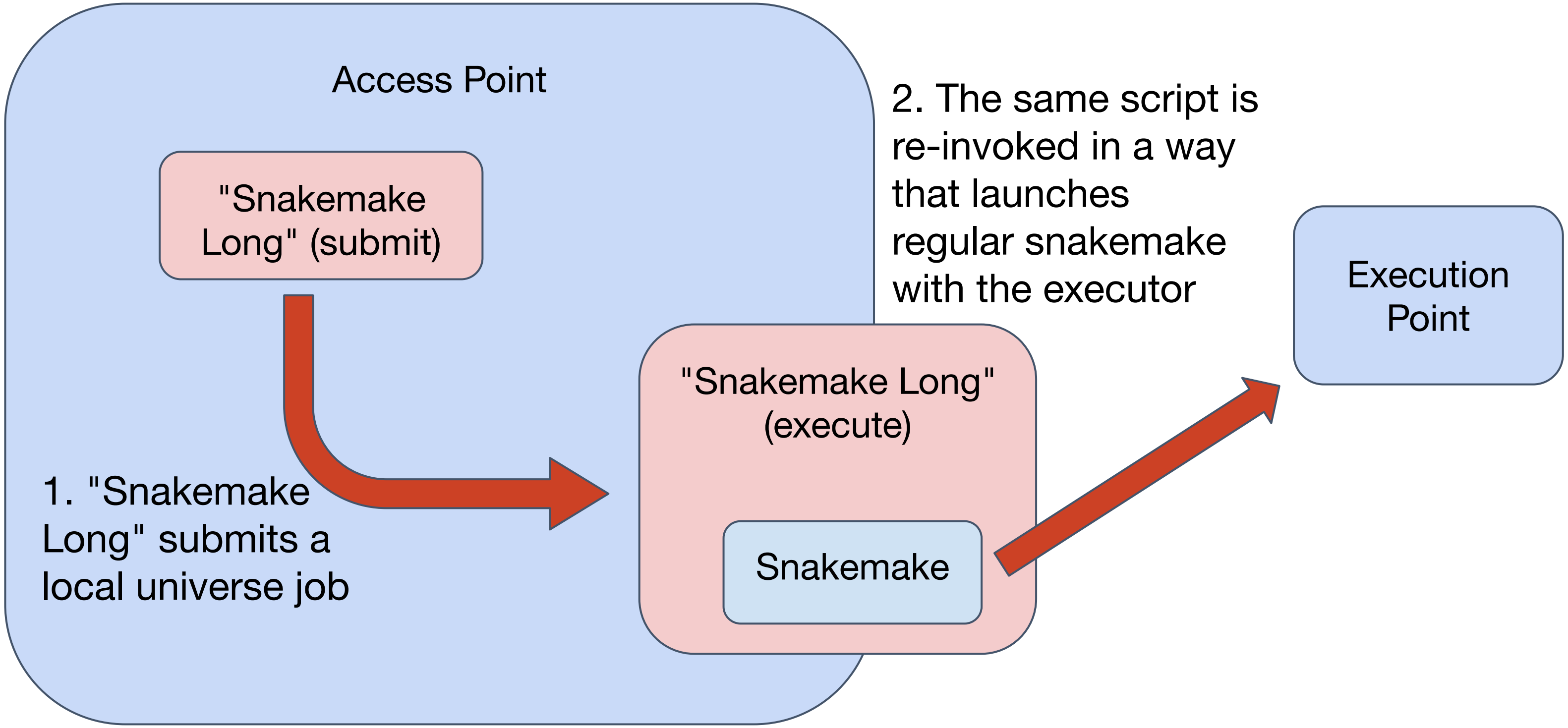
Execution Point

"Snakemake Long"



1. "Snakemake Long" submits a local universe job

"Snakemake Long"



Access Point

"Snakemake Long" (submit)

1. "Snakemake Long" submits a local universe job

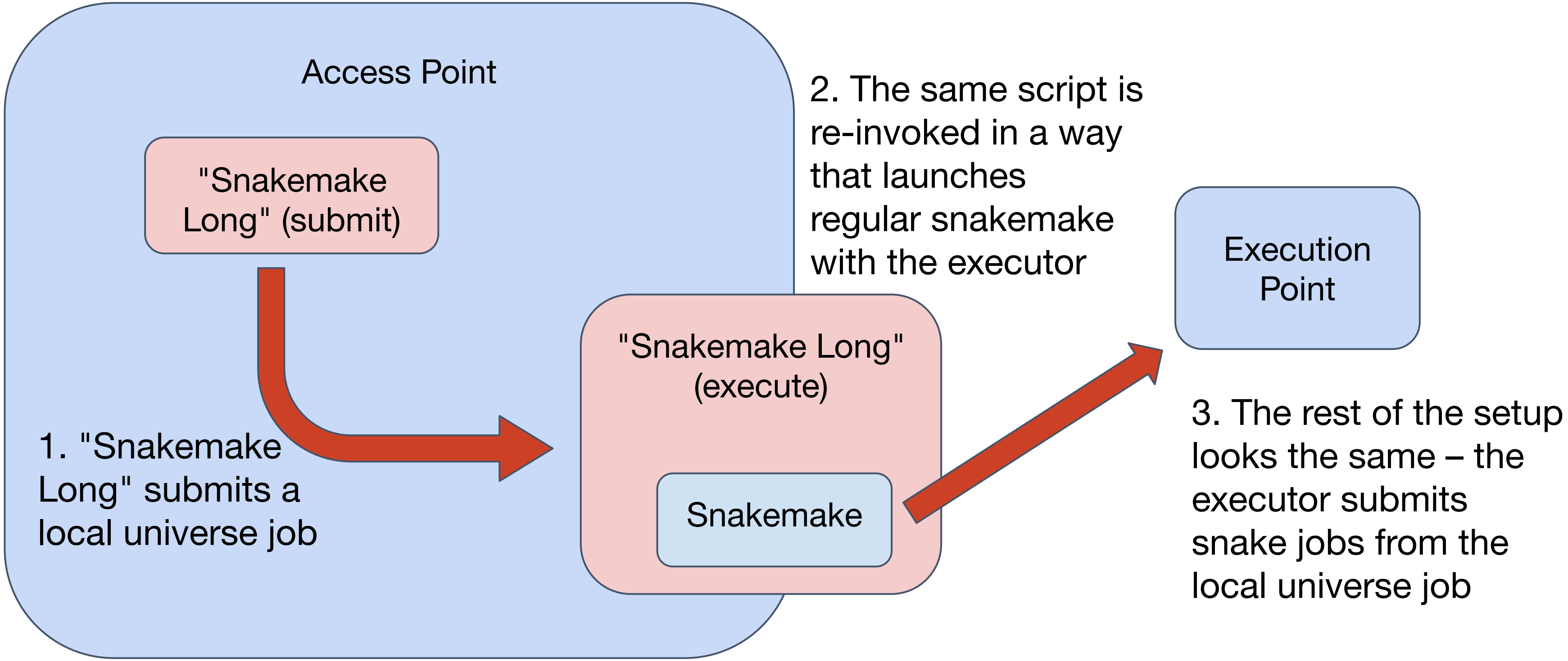
"Snakemake Long" (execute)

Snakemake

2. The same script is re-invoked in a way that launches regular snakemake with the executor

Execution Point

"Snakemake Long"



Access Point

"Snakemake Long" (submit)

1. "Snakemake Long" submits a local universe job

"Snakemake Long" (execute)

Snakemake

2. The same script is re-invoked in a way that launches regular snakemake with the executor

Execution Point

3. The rest of the setup looks the same – the executor submits snake jobs from the local universe job

Next Steps

- The file transfer mechanisms we've discussed are binary – **everything** comes from a shared fs or **nothing** comes from it

Next Steps

- The file transfer mechanisms we've discussed are binary – **everything** comes from a shared fs or **nothing** comes from it
- Snakemake doesn't provide a good way to remap file mounts/names between AP→EP?

Next Steps

- The file transfer mechanisms we've discussed are binary – **everything** comes from a shared fs or **nothing** comes from it
- Snakemake doesn't provide a good way to remap file mounts/names between AP→EP?
- SPRAS jobs still pull algorithm containers within the job – this is bad

Next Steps

- The file transfer mechanisms we've discussed are binary – **everything** comes from a shared fs or **nothing** comes from it
- Snakemake doesn't provide a good way to remap file mounts/names between AP→EP?
- SPRAS jobs still pull algorithm containers within the job – this is bad
- We also want to monitor the inner container's resources

Next Steps

- The file transfer mechanisms we've discussed are binary – **everything** comes from a shared fs or **nothing** comes from it
- Snakemake doesn't provide a good way to remap file mounts/names between AP→EP?
- SPRAS jobs still pull algorithm containers within the job – this is bad
- We also want to monitor the inner container's resources
- I haven't even *touched* getting this to work with my other favorite bird

One key takeaway

"Condor already handles problem X, so just do Y in your workflow" is a dangerous sentiment

There's a lot of value in designing your tools to fit the researcher's needs instead of demanding they fit their workflows to your tools

Just because you have a distributed hammer doesn't mean everything *has* to look like a thumb

Useful Links

SPRAS:

https://github.com/reed-CompBio/spras

HTCondor Snakemake Executor:

https://github.com/htcondor/snakemake-executor-plugin-htcondor

Jannis's Original Snakemake Executor:

https://github.com/jannisspeer/snakemake-executor-plugin-htcondor

Acknowledgement

This material is based upon work supported by the National Science Foundation under Cooperative Agreements DBI-2233968. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

CHTC HTCondor Software Suite PATh