

Matthew Liu¹, Kevin Gott², Zhengji Zhao³
¹UC Berkeley, ²Lawrence Berkeley National Laboratory

ABSTRACT

When creating their parallel job scripts, most NERSC users often disregard the current state of the job queue. Because of this, certain node count and wall time restraints create a much higher wait time compared to the other options.

The goal of this summer project is to explore ways to make the job script generator “smart” by analyzing the queue wait time chart and exploring specific scenarios for VASP. Such a job script generator will allow NERSC users to automate their job launches in a way that cuts their wait time and increases job efficiency.

BACKGROUND INFO

The current job script generator converts straightforward user inputs into a SLURM job script without much flexibility. We explored two possible projects to improve it:

- The **average queue wait time chart** gathers data live to provide users a visualization of the wait time based on the given wall time and node count.

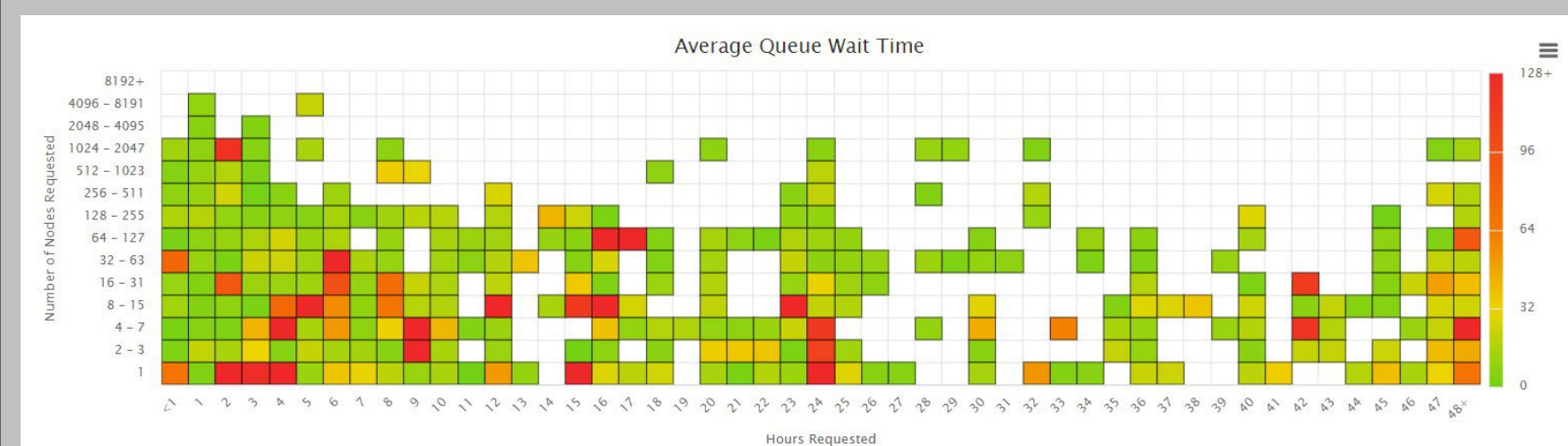


Figure 1: NERSC's average queue wait time chart.
<https://my.nersc.gov/>

- This could be used to make a job script generator that gives users the shortest queue wait time for their jobs.
- VASP** or the Vienna Ab initio Simulation Package is one of the most highly used codes at NERSC.
- VASP's run configuration can be calculated based on input parameters.

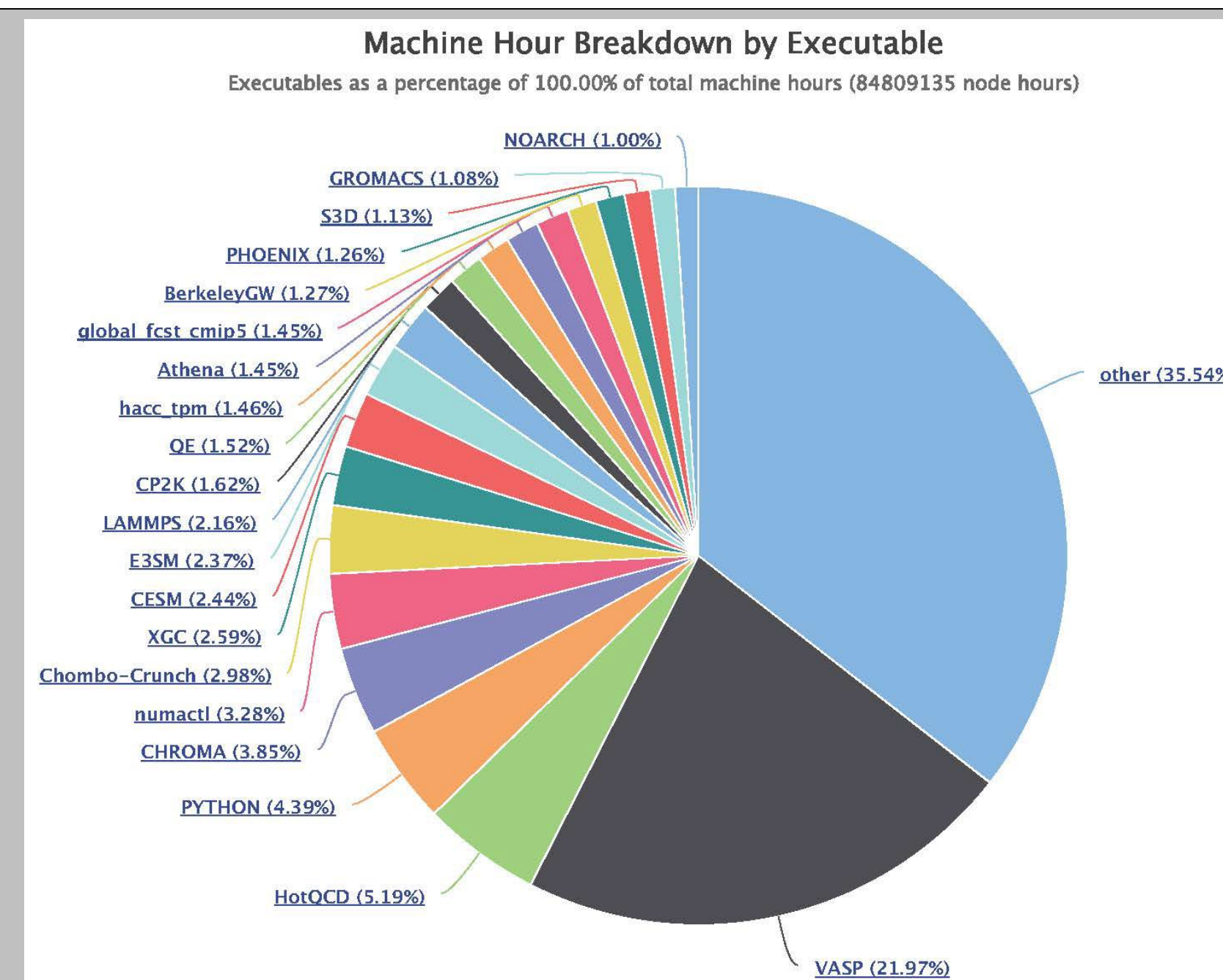


Figure 2: NERSC 2018 Code Usage. VASP usage cover more than 20% of the total machine hour.
 Photo Credit: Zhengji Zhao

There are two main multiprocessing computing models:

- Pure MPI** which allows every parallel process to execute and **hybrid MPI/OpenMP** which appends in OpenMP that allows parallel threads to run on all the data.
- Hybrid MPI/OpenMP is usually recommended for VASP as it outperforms most pure MPI code based on workload.

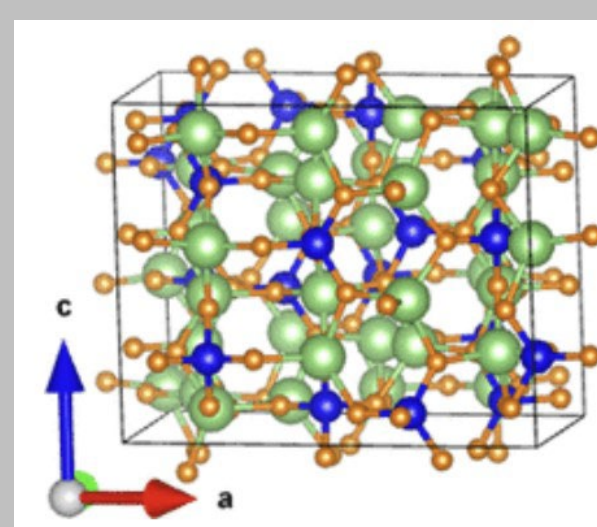


Figure 3: A visualization of an example VASP model
 Photo Credit: Wai-Yim Ching

METHODS

- The original plan was to calculate the lowest 5% wait time settings so that traffic can be distributed when multiple users request the same combination at the same time.
- In order to calculate and estimate the best **wall time & node count** for the job, data had to be pulled from the average queue wait time table and the REST database.
- The problem was that there are tedious API calls that had to be configured in order for this smart generator to work smoothly.

- The team decided to pivot the project to first focus on building a job script generator for VASP.

A new tab was created for the VASP job script generator.

- New options were added e.g. user accessibility, VASP versions, calculating node count and processes count based on k-points and atoms
- Module version** is based on machine type and the binary library chosen
- The formula used to calculate the appropriate **thread count** and **processes count** is as follows:

Number of Processes:

$$\text{nprocs} = \frac{2}{3} \times \langle \text{number of atoms} \rangle \times \langle \# \text{ of kpoint groups} \rangle$$

$$\text{kpoint groups} = \text{kpoints} / 10$$
 Thread Count:

$$< 150, 1 \text{ thread.}$$

$$> 150, 4 \text{ threads. nprocs} *= 4$$

$$> 300, 8 \text{ threads. nprocs} *= 8$$

- In order to ensure the correct output for node count, number processes, etc, hardware system specification were examined.
- The inputs will display error messages if any of the user inputs are not feasible for the specific machine.

RESULT

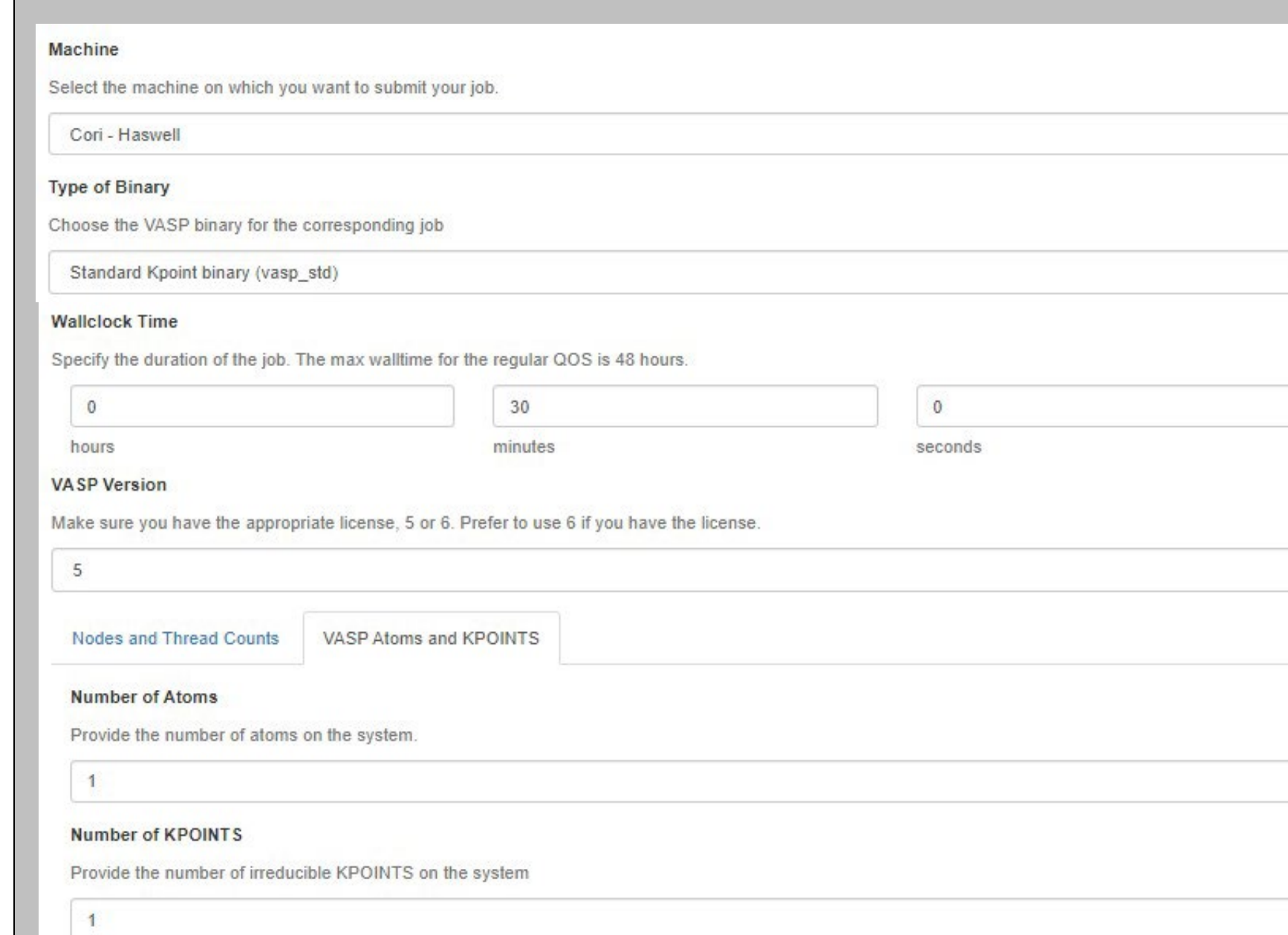


Figure 4: Cropped version of the VASP Job Script Generator.
<https://my.nersc.gov/>

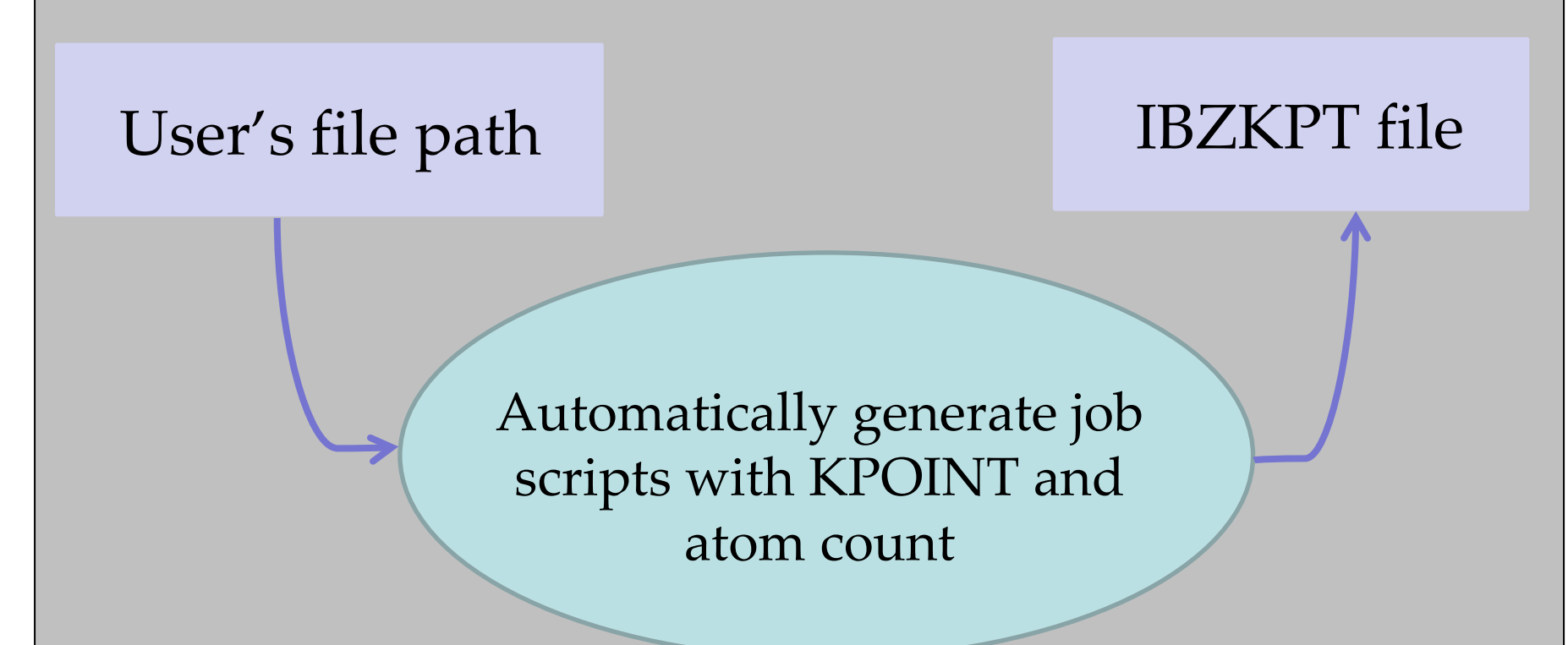
The VASP job script generator is able to successfully catch incorrect inputs specific to machine types, automatically change fields based on user inputs and correctly generate the output file based on different settings.

- Test files were run to verify the accuracy of the output and different settings and error catching for the job script generator were tested.

CURRENT WORK

Next part of the project is accessing the user file directly from the webpage.

- New additions that are in progress include utilizing the **file browser** from my.nersc to directly create and run the job script using the Login nodes.
- The generator is able to extract data such as KPOINTS and atom count directly from the INCAR files.
- The NEWT API commands are able to call commands on Cori and using that, a quick function is able to output the IBZKPT file.



FUTURE WORK

- Combining the UI of the traditional job script generator with the new tab will assist users in not having to input certain fields twice.
- Since the original project was never completed, the next step can be going back to work on the first project proposal
- Discussing with Robin Shao who is also working on the job script generator this summer and merging our projects to simplify the workflow of the overall job script generator.