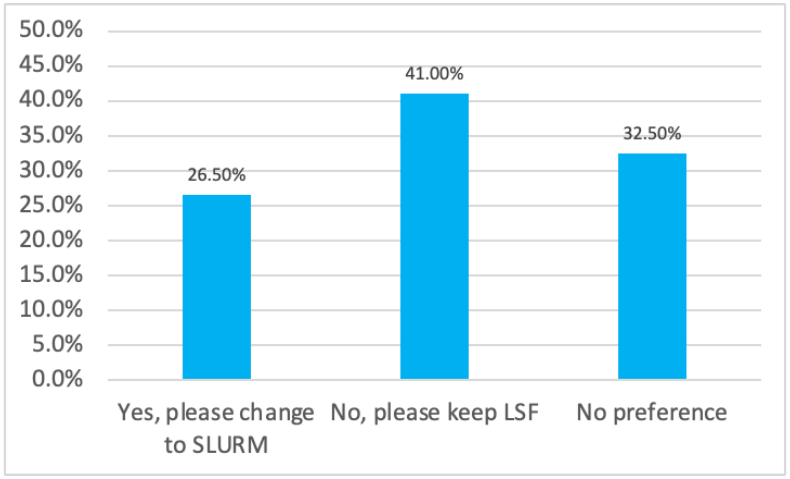
<u>Minerva Job Scheduler User Survey</u>
<u>Scientific Computing and Data</u>
Dec 6th, 2023

The Minerva Job Scheduler User Survey —distributed on Oct. 30th 2023 and ended Nov. 6th 2023 —solicited feedback from 1168 active Minerva users. Of these, 83 users responded (7.1% response rate)

The question we asked:
Do you prefer changing the current Minerva job scheduler LSF to SLURM?
No, please keep LSF (34, 41.0%)
Either one is good for me. No preference (27, 32.5%)
Yes, please change to SLURM (22, 26.5%)



According to the survey result, Minerva will keep using the LSF job scheduler unless there are significant changes in the user workflow, LSF development, Slurm development and budgeting.

—---
Please see below comments from the survey:

<u>Comments from users who chose LSF:</u>

1. I'm used to using LSF; there would be no advantage to using slurm for me and a learning curve to reconfigure my pipelines.
2. Our entire data science and bioinformatics infrastructure is built on LSF. If the HPC teams opts to change to SLURM, we will need to go through considerable rebuild of job submitting infrastructure

3. Does SLURM offer functionality that LSF does not? My use cases for job submission are relatively simple so perhaps I do not have the same pain points as other users, but why require everyone learn a new scheduler and update their scripts at the behest of a few?
4. As mentioned, we have many complicated pipelines that will need quite a bit of work to change to SLURM. LSF works well.
5. I do not have any experience with SLURM but I am regularly using LSF. I have heard of a request to change to SLURM for the first time. If users want to change to SLURM they would need to build a much stronger case with solid arguments for SLURM against LSF. Just wanting to use SLURM is insufficient.
6. Need for migrating existing pipelines (some of which may not support SLURM) and the old adage of "if it isn't broken, don't try to fix it".
7. My workflow(s) use LSF
8. More familiar with the LSF usage. Changing the job scheduler will need additional time to learn and master it.
9. Since I don't have any problem with LSF I would like to avoid having to modify my scripts that the switching would require. If, however, the switch would enable other user to do important things that they can't do with LSF, then be it.
10. While I like the idea of moving on from an expensive closed-source IBD software, I am concerned about the performance overhead of SLURM, migration headaches, and the fact that our processes on LSF are mature now and changing to SLURM means we have to rebuild with new considerations.
11. Translating existing job submission & shell scripts, better familiarity with LSF commands, and experienced some difficulties from a different institution using MATLAB parallel computing with SLURM. I'm not 100% against changing to SLURM though, if it will be more convenient for many users and Scientific Computing.
12. this is a huge problem. you can't just switch because SOME users find it more convenient.   what is the problem with having some of the nodes be slurm and the rest lsf. I cannot emphasize enough that it should not be a unilaterally switch.
13. If I understood the SLURM description correctly, it is node-based, not core-based. It would mean that less users will be served at any given time, which means longer queues.
14. Our lab and many others already have existing pipelines for LSF. I think a better approach would be to keep LSF, and add additional login nodes for running SLURM.
15. I think LSF works well
16. I didn't see the improvement. It will only make work complicated.
17. It might take some time to adapt using a different job scheduler but it's ok if people have problems using LSF.

18. For all the existing users we're going to have to learn a new system when we already learned how to use LSF and this is going to slow down the progress of our ongoing work
19. The potential change to SLURM to benefit some unspecified users is highly detrimental to the rest of the HPC community who has spent significant time and effort to use existing resources. In our particular case, all our pipelines will need to be redesigned to adapt to SLURM. This significantly jeopardizes three funded grants to the lab and one ongoing clinical trial.
20. Familiarity with the system. And if some users want SLURM, why can't we have both options available?
21. LSF works prettey good. Why needs SLURM?
22. Before I know what pros will SLURM bring for me, why would I want to invest weeks of effort to port my scripts from LSF to SLURM???
23. I'm used to the LSF now so don't want take additional effort. But if you want to switch it, could you show us the benefit of SLURM? Thanks, Yanchun
24. I do not want to have to remake my pipelines
25. First, nearly all of our lab's workflows are already optimized for LSF. Perhaps less flexible, the 'cluster mode' of cellranger (the 10X Genomics software suite for processing single cell RNA-Seq data and widely used by MANY labs at ISMMS) is compatible with LSF - this DRAMATICALLY improves performance and computation times.
26. I run lots of small and big jobs on Minerva; LSF provides more flexible controls on jobs as compared to SLURM
27. Learning a new system and updating an entire thesis of code
28. Our lab performs 10X sequencing runs regularly, and uses CellRanger to process these data. CellRanger only works on LSF, so changing to Slurm would radically disrupt our workflows + make our analyses take far longer than they currently do. Please keep as is for this purpose.
29. I regularly use CellRanger on cluster mode, which requires an LSF job scheduler. I have also built infrastructure for my own/labs use around lsf for Snakemake workflow management and would strongly prefer that it remains the same.
30. We heavily rely on cellranger for our single cell sequencing analysis which relies on LSF and is incompatible with SLURM. This is an essential step in our analysis pipelines and we have developed extensive pipelines which rely on this so changes to the scheduler would significantly impair our work.
31. Scripts already in place with LSF. Easier for me to keep going with the same system.
32. I have multiple complex pipelines that we are using and don't want to reconfigure things if it needs significant changes and time.

33. Our laboratory uses CellRanger which is compatible with an LSF submission.
34. All my scripts are developed for LSF. Changing schedulers will require substantial amount of work changeling bash scripts and python snakemake config files. This will disrupt my ongoing work, and I am not aware of any benefits

<u>Comments from users who chose No preference:</u>

1. I've only used LSF so far, and it does the job just fine. I've never used SLURM, but it seems to work fine as well. If I need to switch to SLURM, I would need to change a large chunk of my scripts, but that's not a big deal.
2. I will note, all of my practical experience has come from LSF managed clusters. Theoretically, I am in favor of moving to SLURM. The open source design and wide use of SLURM seem ideal for our HPC. My concerns center around the transition. I am worried about downtime, retraining time (our users are not highly sophisticated computer science people), and the time needed to retool shared scripts like RStudio on the Fly and Jupyter Notebooks.

<u>Comments from users who chose Slurm</u>

1. Familiarity
2. Because I am more familiar with slurm
3. More widespread bioinformatics support for slurm. Easier interfacing with pipeline managers. More human-readable options and better documentation.
4. I have used SLURM extensively and am really comfortable with it. I think creating job submit scripts with SLURM is much more intuitive than LSF
5. much easier to submit jobs and it also has ways of incorporating flags into the sbatch jobs.
6. Open-source, used more commonly, better documentation
7. It is used more widely, so transitions to other HPC after leaving Mount Sinai will be more seamless.
8. switching from university with slurm as its scheduler so scripts are compatible with this
9. It is the more common practice and have better support with external pipeline that developed under SLURM.
10. I used it previously at New York Genome Center. It was more accessible to submit array jobs, and was more manageable in terms of special variables to handle jobs, inputs or outputs.

11. open source and Usage Efficiency
12. More flexibility, more documentation
13. Slurm benefits from wider community support and tool integration. IBM's docs for LSF really only seem useful for an hpc admin
14. SLURM has more functionality/flexibility for scripting
15. I have worked with SLURM before and I found it to be easier to use than LSF. I also like that SLURM is open source.   One concern I have with SLURM is fair sharing of resources among multiple users. I've found documentation from another LSF-to-SLURM transition saying that this might be an issue if not addressed in the implementation: https://scicomp.ethz.ch/wiki/Transition_from_LSF_to_Slurm
16. I've used slurm before on other stacks and its preferential scheduling seems to work better than our LSF system.
17. SLURM is a widely used scheduler. Using it on Minerva would make workflow easier to scale computation across different clusters
18. I have much more experience with SLURM, and it seems to be pretty widely used. However, this is only a slight preference over LSF.
19. Easier to submit jobs more intuitive better explanation of fair sharing I already use it on another server, so would make pipelines and code more transferable
20. The access is easier than the LSF and the platform is more friendly.
21. Open Source
22. SLURM is easier to use, has better documentation, and easier to integrate with bioinformatics pipelines like Snakemake.