Published on *cisTEM* (https://cistem.org)

Home > running time for Refine Pkgs.

running time for Refine Pkgs.

Fri, 05/25/2018 - 15:50 #1

Nicolas

running time for Refine Pkgs.

Hi,

  We have picked particles with cisTEM and are now doing the "Refine Pkgs" step before moving to 2D classification. We are running it on a cluster, but it this steps takes a really long time (estimated time around 1 hour for about 300k particles). It seems a lot and we wanted to check if it's a typical time or if there's something wrong with our setting or submission.

Also, it looks like this particular task is not submitted to the queue of the cluster but is run locally. Is there a way to submit it or any way to make this step faster?

Thanks,

Best,

Nicolas

Fri, 05/25/2018 - 16:27

#2

timgrant

Hi Nicolas,

Hi Nicolas,

Yes this process is run locally on the machine that the GUI is running on. It is basically just cutting out the particles to make a particle stack, and is probably I/O limited. 1 hour for 300k particles does sound slow, unless you have few particles per image (in which case, the limit is probably the time it takes to read in all the movies).

Sorry it is taking so long for you, it is something you only have to do once per particle stack however.

Tim

Fri, 05/25/2018 - 16:49 (Reply to #2)

Nicolas

Hi Tim,

Hi Tim,

Thanks for the quick feedback... Do you think submitting it to the queue could be added to a future release of the software? We have a data \set with more than a million particles and we can't extract the particles picked in cisTEM (we would need +1 day to extract them and the way our machines and cluster are set up, we can't remain connected for a day if we don't inter-act with it). The current solution we came up with is to split it into many smaller sets of particles but it's not a convenient alternative.

Nicolas

Sat, 05/26/2018 - 12:20

#4

timgrant

Hi Nicolas,

Hi Nicolas,

It is certainly something we will consider. However, I am not sure how much faster it will go. It is likely limited by disk access, and so using more processors will not necessarily make it much faster. I think it is going much slower for you than it should (I have not heard of it being this slow for anyone else). If you cut out 1 million particles in another program, is it a lot faster?

Cheers,

Tim

Sat, 05/26/2018 - 19:39 (Reply to #4)

Nicolas

Hi Tim,

Hi Tim,

it seems indeed faster in other programs (Relion). If we try to extract +1 millions directly with cisTEM, the estimation is more than 50 hours, but we haven't actually been able to go to completion because of the way our system is set up for the moment. So as mentioned, we just split it into several smaller sub-groups.

Even is using more processors does not help, being able to submit the job to the queue would already help in our case.

Cheers,

Nicolas

Fri, 07/06/2018 - 19:47

#6

Daniel Asarnow

FWIW, on our cluster at UCSF,

FWIW, on our cluster at UCSF, the interactive node where the GUI runs is much slower (both CPU and network/IO wise) than the actual compute nodes on the queue. If current GUI actions such as creating packages could be submitted, they would be ~10x faster in our case.

Sat, 07/07/2018 - 23:08 (Reply to #6)

timgrant

Hi Daniel,

Hi Daniel,

Thanks for the feedback - do you know why this is the case? Is it the machine, or the program? Can you run an interactive session on a compute node, and run the gui there?

Tim

Sun, 07/08/2018 - 18:41 (Reply to #7)

Daniel Asarnow

It's purely the under-spec

It's purely the under-spec interactive node (and mostly the lack of the 10G network). On our cluster, we can't have interactive access to the compute nodes. We'll eventually add a couple of better interactive nodes and get them on the fast network, but there will always be a limited number of them. I believe at that point *we* wont have any issues, but IMO this situation may be fairly common so it could be worth moving particle extraction / package creation to a subprogram depending on how much work that involves.

With respect to other processing suites, Relion does extraction using an outside call to mpirun and relion_preprocess_mpi, so this problem is avoided. (Unless one tries relion_stack_create, etc. which do use single stacks instead of stack-per-micrograph and suffer from *extraordinary* memory inefficiency). Cryosparc 2 (private beta) likewise submits all of its commands to worker nodes (note - I haven't used it yet).