nature portfolio

Peer Review File

GoToCloud Optimization of Cloud Computing Environment for Accelerating Cryo-EM Structure-Based Drug Design.

Corresponding Author: Dr Toshio Moriya

Version 0:

Reviewer comments:

Reviewer #1

(Remarks to the Author)
The authors in this manuscript present a software package written for structural biology end users to leverage public cloud resources. The authors nicely demonstrate the tradeoff between cost and speed, arriving at specific recommendations for instance type for specific RELION jobs.

The paper is clearly written, I don't have any suggestions for the text or figures.

The only minor comment relates to the code in the online Github repository. Can the authors include a README file with tutorial information on how to use their software? The more information they can provide, the better for new users. Additionally, can the authors provide a license file in the Github repo to clearly indicate under what license the software is released?

Reviewer #2

(Remarks to the Author)
Brief Summary:
The manuscript describes a way to configure Amazon Web Services (AWS) to run Cryo-EM single particle analysis on multiple nodes. The authors use shell scripts to set up the processing environment with AWS products, like Elastic Compute Cloud and ParallelCluster for managing the computing cluster, account management tools for security and billing, and S3 and Elastic File System for storing data. Relion, the data processing software, is compiled and tuned for the hardware on the GPU and CPU instances This is a way to process the large amounts of data (micrograph movies) generated by experiments and in the future could be a way to efficiently run structure based drug design workflows on AWS. The benefit of using AWS or any other cloud computing provider is that users do not need to maintain their own hardware and can dynamically adjust the computational resources depending on the need. Benchmarks for different processing stages are presented and guidance on how the select computing resources based on processing time and cost is provided.

Overall Impression:
As data sizes continue to grow in Cryo-EM and other experimental methods, computational resources and the expertise in managing these resources and associated data processing software need to be more easily accessible to the people performing the experiments. While there are other approaches that also use cloud computing (as cited by the authors), the approach presented in this manuscript supports multiple nodes for a single processing job. The other cited approaches only support single node (but still multi-GPU) or the other approach that also supports multiple nodes has not been updated for several years ("cryoem-cloud-tools", https://github.com/cianfrocco-lab/cryoem-cloud-tools). These wrappers for simplifying the deployment of software and hardware resources lowers the barrier for running these computationally expensive jobs.

Moreover, with the way accounts are set up with this approach, it would be possible for a microscope facility to maintain the data processing software stack and have it available for users who have separate accounts for billing their use of the computational resources. With the granularity of AWS billing, users would really only pay for their usage of AWS resources. And since the authors collaborated with AWS staff, it would be much easier for the microscope facility to maintain and update these scripts to work with updates or improvements in AWS. This approach can centralize the technical aspects of maintaining and optimizing the data processing software, and with optimizing the use of AWS resources, letting the users focus on processing their data.

Specific Comments:

1) Why were the g5 and g4dn instances chosen? Why not p4 (A100) or p5 (H100)? The p series is for general GPU compute. g5 and g4dn seem to be more for machine learning and graphics intensive workflows.
2) Why was the GPU Relion executable built with CUDA_ARCH=75 (from the cmake flags)? For the NVIDIA A10G, it can be 86. Can any benchmarking be done comparing the performance of the A10G GPU with just the CUDA_ARCH=75 optimization and the performance with the CUDA_ARCH=86 optimization? Also, it should be possible to build multiple architecture support into the executable (nvcc does support this, I'm not sure how to modify cmake to do this). Or is this flag separate from optimization? The architecture flag should include some architecture-specific improvements.
3) For completeness, and this can be done in the supplementary materials,
3a) Can you add a sentence to describe the linux distribution used? From the scripts, it looks like Ubuntu 20.04.
3b) What were the sizes (disk space) of the benchmark datasets? That can be added to Table S1. This would give readers an idea of the dataset sizes.
4) Did the reported costs include storage costs (S3, EFS, Lustre)? From the spreadsheet, it looks like it is the processing costs only. Can you provide a breakdown of the total costs in terms of processing time (compute) and storage?
5) There are hardcoded S3 bucket URLs in scripts (e.g. https://kek-gtc-master-s3-bucket.s3.ap-northeast-1.amazonaws.com/gtc_efs_setting.json in gtc_setup_gotocloud_environment.sh). Is it intended for the scripts to have hardcoded URLs for your S3 bucket? Are the scripts useable by other people to set up their own clusters? Also, the S3 URLs to be publicly accessible?
6) How long does it take to transfer data to the S3 bucket? Are you transferring from Japan to an AWS region in the United States?
7) Also, how long does it take to transfer data from the S3 bucket to the Amazon FSx Lustre filesystem?

Reviewer #3

(Remarks to the Author)
The manuscript concerns the computational requirements of Single Particle Analysis for cryoEM. These can be high, and it is certainly true that structural biologists often use inappropriate setups and hardware, and may do more processing than necessary. Target resolution is also a key quantity, and achieving the highest resolution possible is not always necessary for the biological question.

Many facilities and groups have sought to optimise the use of compute resources. Some efforts have been published, most not. Relion has been benchmarked several times before. Cianfrocco (ref 16) highlighted the benefits of spot instances on AWS in 2015. The question is whether this article provides any additional insights or a practical way for others to manage their resources.

GoToCloud supports multi-node processing using a selection of common cryoEM software via AWS pcluster. It manages the hardware selection, dependencies, and cryoEM software installation. It also handles configuration with respect to AWS Regions. It would appear to be a good solution for the authors, but it is unclear to me to what extent the code or the article are useful to others.

Pre-installation of a range of executables is handled by a management account. It was unclear at first whether the authors were providing a service to the community, but I believe they are only providing the tools to do this. It is not even clear if it is being used at the Photon Factory. Maintaining up-to-date software is a headache for all facility and compute managers. Software gets rapidly out-of-date, as shown by their usage of Relion 3 and 4 (version 5 is now out).

Their benchmarks show that the cost/performance can be optimised for each job type. They choose a criterion of closest to the origin of a cost vs time graph, and I think that is reasonable. They show that this is dependent on instance type, and will need to be updated for as new hardware comes along. More urgently, I would assume that it is dependent on the dataset. They have only benchmarked this calculation on a single example (nitrite reductase) and so the variation in their conclusions is not tested. They do not describe what would happen in normal usage. Does this benchmarking have to be done on each new dataset in order to determine the best configuration? Is the user expected to do this? Can anything be inferred from short runs (probably not as Relion searches tend to be adaptive)?

The test against resolution was not what I was expecting. They record the computational requirements as the structure solution progresses through the different stages, rather than testing against adjustable parameters (such as number of particles picked, search parameters, etc). Possibly the only practical decision to be made is how many cycles of particle polishing to do. Here, the conclusion is a bit strange. After emphasizing the substantial cost of these additional cycles, they conclude it is worth doing anyway. They suggest that the modest resolution gain will help model building, although this is not actually tried. Overall, this test seems a bit pointless, unless they are going to demonstrate some kind of stopping condition. Again, this test is done on a single benchmark, so it is not clear how general it is.

The core of GoToCloud seems to be a set of GTC configuration scripts. These are available on github, and are a set of bash scripts. I could see no documentation there, and I think it would be very difficult for anyone else to set up GoToCloud, let alone adapt it or update it. There is some explanation in the Supplementary Methods which helps to give the idea, but not enough for a 3rd party to adopt this solution.

Some minor comments:

1. Figures 2 to 5 show the same plots for 4 different job types. I don't think this is necessary and some space could be saved.
2. Figure 1a contains a typo - it should be UCSF Chimera(X).

In conclusion, although the work is technically correct, I don't think there is much novelty in this article. The benchmarks and conclusions are specific to the two examples studied, and there have been other benchmarks published. If it is meant to be a practical solution, then there are many questions around how someone would deploy the GoToCloud system, and how users would use it for their datasets. I cannot recommend publication.

Author Rebuttal letter:

1

2 Response to Reviewers
3
4 Reviewers' comments:
5
6 Responses to Reviewer #1
7 C1-1) The authors in this manuscript present a software package written for structural
8 biology end users to leverage public cloud resources. The authors nicely demonstrate the
9 tradeoff between cost and speed, arriving at specific recommendations for instance type
10 for specific RELION jobs.
11 The paper is clearly written, I don't have any suggestions for the text or figures.
12
13 The only minor comment relates to the code in the online Github repository. Can the
14 authors include a README file with tutorial information on how to use their software?
15 The more information they can provide, the better for new users. Additionally, can the
16 authors provide a license file in the Github repo to clearly indicate under what license the
17 software is released?
18
19 Response 1-1) Thank you for your encouraging feedback on our manuscript and for
20 taking the time to review the associated GitHub repository. We appreciate your valuable
21 suggestion regarding the inclusion of a README file and a license file. We agree that
22 providing additional information for new users is crucial to significantly enhance the
23 usability and accessibility of our code for the research community. Therefore, we have
24 included both a README file and a license file in the GitHub repository
25 (https://github.com/KEK-SBRC-CryoEM/gotocloud) and added the following sentences
26 in line 529-534 of the main text: "The README file in the GitHub repository provides
27 step-by-step instructions on how to use the GoToCloud scripts for setting up the
28 GoToCloud platform and creating the instances on user's AWS account. It also includes
29 links to websites where readers can find detailed versions of these procedures,
30 comprehensive tutorials on using the GoToCloud scripts, and installation instructions for
31 the main software supported by the GoToCloud platform." The license file is for the GNU
32 GENERAL PUBLIC LICENSE Version 3, which clearly outlines the terms under which
33 our GoToCloud scripts can be used, modified, and distributed. Thank you once again for
34 your feedback and for helping us improve the accessibility and usability of our software.
35

1
36 Responses to Reviewer #2
37
38 Brief Summary:
39 The manuscript describes a way to configure Amazon Web Services (AWS) to run Cryo-
40 EM single particle analysis on multiple nodes. The authors use shell scripts to set up the
41 processing environment with AWS products, like Elastic Compute Cloud and
42 ParallelCluster for managing the computing cluster, account management tools for
43 security and billing, and S3 and Elastic File System for storing data. Relion, the data
44 processing software, is compiled and tuned for the hardware on the GPU and CPU
45 instances This is a way to process the large amounts of data (micrograph movies)
46 generated by experiments and in the future could be a way to efficiently run structure
47 based drug design workflows on AWS. The benefit of using AWS or any other cloud
48 computing provider is that users do not need to maintain their own hardware and can
49 dynamically adjust the computational resources depending on the need. Benchmarks for
50 different processing stages are presented and guidance on how the select computing
51 resources based on processing time and cost is provided.
52
53 Overall Impression:
54 As data sizes continue to grow in Cryo-EM and other experimental methods,

computational resources and the expertise in managing these resources and associated data processing software need to be more easily accessible to the people performing the experiments. While there are other approaches that also use cloud computing (as cited by the authors), the approach presented in this manuscript supports multiple nodes for a single processing job. The other cited approaches only support single node (but still multi-GPU) or the other approach that also supports multiple nodes has not been updated for several years (“cryoem-cloud-tools”, https://github.com/cianfrocco-lab/cryoem-cloud-tools). These wrappers for simplifying the deployment of software and hardware resources lowers the barrier for running these computationally expensive jobs.

Moreover, with the way accounts are set up with this approach, it would be possible for a microscope facility to maintain the data processing software stack and have it available for users who have separate accounts for billing their use of the computational resources. With the granularity of AWS billing, users would really only pay for their usage of AWS resources. And since the authors collaborated with AWS staff, it would be much easier for the microscope facility to maintain and update these scripts to work with updates or improvements in AWS. This approach can centralize the technical aspects of maintaining

and optimizing the data processing software, and with optimizing the use of AWS resources, letting the users focus on processing their data.

Thank you for your accurate and concise summary of our current work. Your expert comments on AWS were particularly valuable in enhancing the quality of our current paper.

Specific Comments:

C2-1) Why were the g5 and g4dn instances chosen? Why not p4 (A100) or p5 (H100)? The p series is for general GPU compute. g5 and g4dn seem to be more for machine learning and graphics intensive workflows.

Response 2-1) Thank you for pointing out that we have omitted an explanation of why we chose G4dn and G5 over P4 and P5. We added the following sentences in line 198-207 of the main text: "The P4 (p4d.24xlarge) and P5 (p5.48xlarge) instances were also considered, but we chose the G4dn and G5 instance types over P4 and P5 primarily because of their flexibilities in GPU card numbers and hourly rates that allow us to optimize number of GPUs (i.e. less than 8 GPU cards) and so reduce the cost for each RELION job type. While P4 and P5 instance types support only the 8 GPU configuration, G4dn and G5 instance types offer configurations with 1, 4, and 8 GPU cards. Additionally, the hourly rates for P4 (32.7726 USD) and P5 (98.32 USD) instances are significantly higher than those for G4dn (USD 7.824 for g4dn.metal) and G5 (USD 16.288 for g5.48xlarge) instances with 8 GPU cards. Therefore, finer adjustments for cost performance are achievable with G4dn and G5."

C2-2) Why was the GPU Relion executable built with CUDA_ARCH=75 (from the cmake flags)? For the NVIDIA A10G, it can be 86. Can any benchmarking be done comparing the performance of the A10G GPU with just the CUDA_ARCH=75 optimization and the performance with the CUDA_ARCH=86 optimization? Also, it should be possible to build multiple architecture support into the executable (nvcc does support this, I’m not sure how to modify cmake to do this). Or is this flag separate from optimization? The architecture flag should include some architecture- specific improvements.

Response 2-2) We are grateful for your suggestion of additional optimization possibilities for the G5 instance types equipped with NVIDIA A10G Tensor Core GPUs. For this, we

additionally built a RELION4.0 executable with CUDA_ARCH=86 and conducted benchmarks for Refine3D, Class3D, and Class2D using the G5 instances. The results showed clear improvements in execution times and costs for Refine3D, some improvement for Class3D, and no improvement for Class2D. The improvements become more pronoun as the number of nodes increases. Accordingly, we added the “Results of a further optimization for the G5 instance types” subsection in line 306-315 of the main text as follow:

“Results of a further optimization for the G5 instance types:

A further optimization for the G5 instance types equipped with NVIDIA A10G Tensor Core GPUs were tempted. We built another RELION excitable using the cmake

configuration optimal for the compute capability of the CUDA architecture of A10G (i.e., “-DCUDA_ARCH=86”), in addition to the one which is optimal for the G4dn with NVIDIA T4 Tensor Core GPU (i.e., “-DCUDA_ARCH=75”), and conducted benchmarks for Refine3D, Class3D, and Class2D using the G5 instance types. The results showed clear improvements in execution times and costs for Refine3D, slight improvement for Class3D, and no improvement for Class2D (Fig. S6). The improvements become more pronoun as the number of nodes increases.“
Also, figure S6 has been included in the Supplemental Information to present these results, and raw data has been appended to the Refine3D, Class3D, and Class2D sheets in Supplemental Data 2. To provide the details on compilation settings and the types of RELION jobs applied, we added the "EXE04_GPU" entry in Supplemental Data 1.

C2-3) For completeness, and this can be done in the supplementary materials,
C2-3a) Can you add a sentence to describe the linux distribution used? From the scripts, it looks like Ubuntu 20.04.

Response 2-3a) Thank you for pointing out that the description of the Linux distribution was missing. We added the following sentence in line 174-175 of the Supplemental Information: "Currently, the Linux distribution of this plucster instance is set to Ubuntu 20.04 LTS (Long Term Support) in the pcluster configuration file.”

C2-3b) What were the sizes (disk space) of the benchmark datasets? That can be added to Table S1. This would give readers an idea of the dataset sizes.

Response 2-3b) We appreciate your suggestion. We have added the sizes of the benchmark datasets (EMPIAR-10581: 1.0 TB (MRC), EMPIAR-10641: 373.1 GB (TIFF)) to Table S1 as requested.

C2-4) Did the reported costs include storage costs (S3, EFS, Lustre)? From the spreadsheet, it looks like it is the processing costs only. Can you provide a breakdown of the total costs in terms of processing time (compute) and storage?

Response 2-4) We appreciate you bringing the significance of AWS storage costs to our attention. We agree that this topic is of great interest to readers. As you pointed out, the reported figures were the processing costs only. To address this point, we added a breakdown of all the associated costs and calculated the “Grand Total Cost”, which includes the storage costs, in Supplemental Data 2. The columns for the costs of the pcluster head node, Lustre, S3 and EFS associated to each benchmark run are added. Since the pricing of the S3 and EFS provided by AWS are listed as a monthly rate per GB (USD/GB/month), we first checked the billing statement of our AWS account for the month when we conducted the benchmark tests (February 2023), and then extracted the costs associated to the S3 and EFS. Newly added the “Pricing” sheet in Supplemental Data 2 provides these costs. For the cost calculation of each RELION job execution, these costs in the month are further multiple by the ratio of the process time (in hours) relative to the period of the month. For this, the following paragraph were added to the main text (line 217-224): "As reference for readers, the Supplemental Data 2 also contains a breakdown of all the associated costs as well as a “grand total cost”, including the storage costs. Since the pricing of the S3 and EFS provided by AWS are listed as a monthly rate per GB (USD/GB/month), the costs associated to the S3 and EFS were extracted from the billing statement of our AWS account for the month (February 2023) when the benchmark tests were conducted (provide in the “Pricing” sheet). For the calculation of the ground total cost for each RELION job execution, these storage costs in the month are further multiple by the ratio of the process time (in hours) relative to a month.”

C2-5) There are hardcoded S3 bucket URLs in scripts (e.g. https://kek-gtc-master-s3-bucket.s3.ap-northeast-1. amazonaws.com/gtc_efs_setting.json in gtc_setup_gotocloud_environment.sh). Is it intended for the scripts to have hardcoded URLs for your S3 bucket? Are the scripts useable by other people to set up their own clusters? Also, the S3 URLs to be publicly accessible?

Response 2-5) Thank you for taking the time to review the GoToCloud scripts and

finding the problem. The command interface of gtc_setup_gotocloud_environment.sh is modified in the GoToCloud GitHub. Now, the command has an option “-o OWN_SHARED_S3_BUCKET”, with which user can specify the path to a shared S3 bucket of user’s own GoToCloud platform. The default path is still the hardcoded URLs of the shared S3 bucket of our GoToCloud platform.

C2-6) How long does it take to transfer data to the S3 bucket? Are you transferring from Japan to an AWS region in the United States?

Response 2-6) We appreciate you bringing up another important aspect of AWS regarding data transfer speed. We agree that this topic is of great interest to readers as well. We have been constantly transferring cryo-EM data from KEK (Tsukuba, Japan) to S3 buckets in the Tokyo AWS region. Also, we have less-regularly transferred cryo-EM data from KEK to the US East (Northern Virginia) AWS Region for our developmental purpose like the benchmark tests of the current work. To answer your questions, we measured the times for the data transfers (three times each using 1.0 TB EMPIAR-10581dataset) as follow:

(1) The Lustre storage at KEK to a S3 bucket in the Tokyo AWS region:
Transfer time: 77.91 ± 1.39 minutes
Command line: time aws --profile=kek903_kek-cs-scheme s3 sync /lustre1/emdata/gotofly/gtc_benchmark/empiar-10581/10581/data/EMD_0731_181122 s3://kek-cs-scheme-325321040781-empiar10581-transfer-tokyo/EMD_0731_181122

(2) The Lustre storage at KEK to a S3 bucket in the US East (Northern Virginia) AWS region:
Transfer time: 235.18 ± 2.48 minutes
Command line: time aws --profile=kek903_kek-cs-scheme s3 sync /lustre1/emdata/gotofly/gtc_benchmark/empiar-10581/10581/data/EMD_0731_181122 s3://kek-cs-scheme-325321040781-empiar10581-transfer-us/EMD_0731_181122

The data transfer times are always similar to the above and match with our experiences over a few years of using AWS. To reflect these results, the following sentences were added in line 178-187 of the Supplemental Information: “As examples, the averages of times to transfer a 1.0 terabyte (TB) input dataset (EMPIAR-1058) for three times were as follow: (1) 77.91 ± 1.39 minutes from KEK (Tsukuba, Japan) to S3 buckets in the

Tokyo AWS region using “time aws s3 sync” command on an on-premise computer at KEK, (2) 235.18 ± 2.48 minutes from KEK to the US East (Northern Virginia) AWS Region using the same method as (1), and (3) 90.26 ± 1.90 minutes from a S3 bucket to the Amazon FSx Lustre filesystem within the US East (Northern Virginia) AWS region using “time cp -r” command on the header node of the pcluster associated to the FSx. The reported time (3) is averages of the execution time differences between the copy command before preloading (148.43 ± 1.90 minutes) and after preloading (58.17 ± 0.00 minutes).” Here, the description of (3) is for Response 2-7.

C2-7) Also, how long does it take to transfer data from the S3 bucket to the Amazon FSx Lustre filesystem?

Response 2-7) Thank you again for pointing out a vital data transfer for exploiting AWS ParallelCluster. To answer your question, we measured the data transfer time from a S3 bucket to the Amazon FSx Lustre filesystem within the US East (Northern Virginia) AWS region (three times using 1.0 TB EMPIAR-10581 dataset) and obtained the average data transfer time of 90.26 ± 1.90 minutes. Again, the data transfer time is always similar to this and match with our experiences.

The following command line was executed after deleting the source directory (/fsx/10581/Micrographs/) from the AWS FSx Lustre, and then importing the metadata of the files in this source directory from the S3 bucket associated to the pcluster. This way, we ensured the lfs hsm_state of all files in the source directory are “released exists archived”, meaning the preloading (i.e., data transfer time from a S3 bucket to the Amazon FSx Lustre filesystem) is necessary to start using the files on the AWS FSx Lustre.

$ time cp -r /fsx/10581/Micrographs /fsx/20240810_measure_copy_before_preload

The average data transfer time was 148.43 ± 1.90 minutes (executed three times). Then, again executed the same copy command as follow.

$ time cp -r /fsx/10581/Micrographs /fsx/20240810_measure_copy_after_preload

The average data transfer time was 58.17 ± 0.00 minutes (executed three times). The reported preloading times above is an average of the execution time differences between

copy before preloading and after preloading. Please refer Response 2-6 for the addition of sentences associated this response.

Responses to Reviewer #3

7

C3-1) The manuscript concerns the computational requirements of Single Particle Analysis for cryoEM. These can be high, and it is certainly true that structural biologists often use inappropriate setups and hardware, and may do more processing than necessary. Target resolution is also a key quantity, and achieving the highest resolution possible is not always necessary for the biological question.

Response 3-1) Thank you for expressing your agreement regarding the target resolution.

C3-2) Many facilities and groups have sought to optimise the use of compute resources. Some efforts have been published, most not. Relion has been benchmarked several times before. Cianfrocco (ref 16) highlighted the benefits of spot instances on AWS in 2015. The question is whether this article provides any additional insights or a practical way for others to manage their resources.

Response 3-2) We thank the reviewer for pointing out the relevant research in this area. The aim of our current work is on building a practical platform for computations required in the cryo-EM field using the latest cloud computing technologies and currently available services provided by AWS, rather than how to run RELION on AWS optimal way. As reviewer #2 kindly summarized the essence of our paper concisely in “Overall Impression”, our approach can centralize the technical aspects of maintaining and optimizing the data processing software, and with optimizing the use of AWS resources, letting the users focus on processing their data. The architecture design and the utilization of spot instances described in the Cianfrocco (ref 16) were nicely done using the technologies and services available at that time. Here, we devised more recent technologies and services provide by AWS. For example, we use AWS ParallelCuster instead of STARcluster. This allows user to easily use the multiple latest AWS instance types (e.g. G5, G4dn, C6i) within a single workflow of Cryo-EM SPA. Cianfrocco (ref 16) described the usage of a single EC2 instance type “r3.8xlarge” but there is no mention about how to use the Amazon Machine Image (AMI) for the usage of multiple AWS EC2 instance types within a single workflow. Also, the FSx for Lustre is used in GoToCloud as the storage whose access speed is significantly faster than elastic block storage (EBS) used in Cianfrocco (ref 16). Therefore, to clarify our aim, we added the following sentences in line 101-105 of the main text: “The aim is on establishing a practical computational infrastructure for the cryo-EM field using the latest cloud computing technologies through a publicly available service for both academia and industry. Our

8

approach can centralize the technical aspects of maintaining and optimizing the data processing software, and with optimizing the use of AWS resources, letting the users focus on processing their data.”

C3-3) GoToCloud supports multi-node processing using a selection of common cryoEM software via AWS pcluster. It manages the hardware selection, dependencies, and cryoEM software installation. It also handles configuration with respect to AWS Regions. It would appear to be a good solution for the authors, but it is unclear to me to what extent the code or the article are useful to others.

Response 3-3) Thank you for pointing out an ambiguity in our writing regarding the usefulness of GoToCloud. There are already users using the GoToCloud platform. We have been receiving feedbacks from them saying that the GTC scripts and the online documents including step-by-step instructions, which we have prepared for the participants of the GoToCloud hands-on workshop, are easy to use and helpful. The users became able to build GoToCloud platform instances for their own cryo-EM SPA projects by themselves after a single online hands-on workshop with us. The online documents were in Japanese but the translation to English is in progress (please also refer Response 1-1). To clarify the above point, we added the following sentences in line 168-176 of the main text: “A manual procedure building a GoToCloud platform instance typically takes approximately half a day to a full day even for the developers who have well familiarized with the steps. With the GTC scripts and online documentations of step-by-step

instructions (see the Code availability section), this task can be completed in 30-60 minutes for users at any level of familiarity with the AWS system, making the GTC scripts highly valuable. Moreover, since most of the time spent is just waiting for the command executions, the actual time spent by the user is even shorter. In addition, by carefully designing the specification of the GTC scripts, the possibility of user making mistakes has been minimized."

C3-4) Pre-installation of a range of executables is handled by a management account. It was unclear at first whether the authors were providing a service to the community, but I believe they are only providing the tools to do this. It is not even clear if it is being used at the Photon Factory.

Response 3-4) Thank you for your interest in our service to the community in the GoToCloud project. We have been providing not only the tool but also services to the

community by actively supporting the introduction of the GoToCloud platform to users' own AWS accounts and the establishment of their supporting environments. Furthermore, since July 2021, we have conducted several GoToCloud hands-on workshops for both external users from academia and industry, with 1-4 participants each. As explained in Response 3-3, there have already been both academic and industrial users utilizing the GoToCloud platform for over a year, and internal use at our facility (i.e., the Photon Factory) has also commenced.

C3-5) Maintaining up-to-date software is a headache for all facility and compute managers. Software gets rapidly out-of-date, as shown by their usage of Relion 3 and 4 (version 5 is now out).

Response 3-5) We appreciate your understanding of the challenges faced by facility and compute managers in keeping software up-to-date. The RELION5.0-beta (an online release announcement on October 27, 2023) has been installed and can be used already in the GoToCloud platform but this version is not set to default. Given that an unstable version is likely to be superseded by a stable release soon, it is inefficient to invest development efforts in it because repeating the optimization for each new software release requires considerable effort. Therefore, our basic policy is to support only the latest "stable" version as the default in the GoToCloud platform. Our plan is to make RELION-5.0 default as soon as the stable release is announced, conduct large-scale benchmark tests similar to ones provided in our current paper, and publish the results on our WEB site. To clarify this, we added the following sentences in line 459-465 of the main text: "RELION5.0-beta (an online release announcement in October 2023) has been also pre-installed and can be used already in the GoToCloud platform but this version is not set to default currently (RELION4.0 is current default). Given that an unstable version is likely to be superseded by a stable release soon, it is inefficient to invest development efforts in it because repeating the optimization for each new software release requires considerable effort. Therefore, our basic policy is to support only the latest stable version as the default in GoToCloud."

C3-6) Their benchmarks show that the cost/performance can be optimised for each job type. They choose a criterion of closest to the origin of a cost vs time graph, and I think that is reasonable. They show that this is dependent on instance type, and will need to be updated for as new hardware comes along. More urgently, I would assume that it is dependent on the dataset. They have only benchmarked this calculation on a single

example (nitrite reductase) and so the variation in their conclusions is not tested. They do not describe what would happen in normal usage. Does this benchmarking have to be done on each new dataset in order to determine the best configuration? Is the user expected to do this? Can anything be inferred from short runs (probably not as Relion searches tend to be adaptive)?

Response 3-6) Thank you for pointing out that we missed to describe what would happen in normal usage. In answer to the questions, similar benchmark tests do not need to be performed for each dataset, and users are not expected to conduct benchmark tests to find the optimal parallel settings for each of their datasets. Performing benchmarks on a test case provides sufficient assurance that the overall performance for a workflow in general use cases will achieve adequate level of optimality (please see below for the rationale). We agree that this discussion is important, and so, the following paragraph is added in

line 383-400 of the main text. “An important consideration is the generality of proposed optimal parallel computing settings for each RELION job type: here, we argue a rationale for which benchmark tests do not need to be performed for each dataset and so a test case using a single dataset should provide sufficient assurance that the overall performance of a workflow in normal usage will achieve adequate level of optimality. Generally, for a given job type (e.g. Refine3D), a “absolute” processing time and cost using a specific set of parallel computing settings varies depending on dataset and processing step (where Refine3D is used), because the computational loads are different while the computational capability remains the same. On the other hand, “relative” relationship between two sets of parallel computing settings should remain the same regardless of the computational loads (e.g., “A” settings is always faster and cheaper than “B” settings), because, for each relationship, the computational load is expected to remain relatively constant (possibly with only minor fluctuations due to the use of random numbers by the algorithms) while the computational capability of two sets of parallel computing settings are different. Thus, by fixing the required computational load to a practical amount (not too small) and optimizing the parallel computing settings (i.e., benchmarking for a specific dataset and job), the optimal choice is expected to be nearly identical for a particular job type regardless of dataset and processing stage at a practical level. “

C3-7) The test against resolution was not what I was expecting. They record the computational requirements as the structure solution progresses through the different stages, rather than testing against adjustable parameters (such as number of particles picked, search parameters, etc). Possibly the only practical decision to be made is how

many cycles of particle polishing to do. Here, the conclusion is a bit strange. After emphasizing the substantial cost of these additional cycles, they conclude it is worth doing anyway. They suggest that the modest resolution gain will help model building, although this is not actually tried. Overall, this test seems a bit pointless, unless they are going to demonstrate some kind of stopping condition. Again, this test is done on a single benchmark, so it is not clear how general it is.

Response 3-7) Thank you for pointing out that the motivation of the test against resolution was not clearly explained. While optimization of adjustable parameters of each algorithm type is important, our purpose of this test is different: it is a case study to give an example of how the processing time and cost of a practical usage increases when one tries to improve resolution by doing additional processes on the GoToCloud platform. Our motivation was to find out if the relationships between the processing time and cost relative to the resolution improvement are more like linear or exponential using a typical SPA workflow. We believe that this example provides users a useful clue to decide a minimum necessary resolution and for this up to which process stage they should employ for a given research purpose. To clarify this point, we added the following sentences in line 319-323 of the main text: “The motivation of this case study was to find how the processing time and cost of a practical usage increases when one tries to improve resolution by doing additional processes on the GoToCloud platform and if the relationships between the processing time and cost relative to the resolution improvement are more like linear or exponential using a typical SPA workflow.”

Thank you for your suggestion about Polish; we also think that a main practical decision to be made is how many cycles of Polish to run. Therefore, we added the following sentences in line 355-358 of the main text: “This result indicates that one should consider omitting jobs in the later stage of the SPA processing if the resolution sufficient for research purpose is achievable without them. Specifically, employing a smaller number of Polish cycles is effective to reduce the processing time and cost.”

We appreciate your pointing out that the statement of “the modest resolution gain will help model building” is misleading and not verified and we agree with this point. Therefore, we removed the following sentences from the last paragraph of the “Relationship between cost and resolution improvement in the Cryo-EM SPA workflow” section in Result of the main text. “The appearances of the side chains strongly affect the accuracy of atomic coordinate modeling and contribute to substantial improvements in the modeling and associated statistical values. Therefore, in the

resolution range of around 2.0 Å, even if the improvement of the resolution is only in the second decimal place, it should not be undervalued.”.

Also, we agree that it is important to demonstrate some kind of stopping condition,

especially for the automation of the SPA workflow. However, this is not the scope of our current work. Therefore, we modified the following sentence in line 413-416 of the main text. “As the processing result of the streptavidin dataset indicated, an important issue for the automation is to define the stopping condition of the workflow to achieve a high level of practicality and should be addressed in this future study.”
Because of a similar rationale explained in Response 3-6, performing a single benchmark on a test case should provide a sufficient clue to readers about an expected trend in the increases of total process time and cost relative to resolution improvement for a typical processing workflow. However, again, validating this argument is beyond the scope of this paper, it should be addressed in a future work for the full automation of the cryo-EM SPA workflow.

C3-8) The core of GoToCloud seems to be a set of GTC configuration scripts. These are available on github, and are a set of bash scripts. I could see no documentation there, and I think it would be very difficult for anyone else to set up GoToCloud, let alone adapt it or update it. There is some explanation in the Supplementary Methods which helps to give the idea, but not enough for a 3rd party to adopt this solution.

Response 3-8) Thank you for reviewing the associated GitHub repository and for helping us improve the accessibility and usability of our current work. As described in Response 1-1, we added two documentations to the GitHub: (1) README file with tutorial information on how to use the GTC configuration scripts, and links to our web site providing more elaborated version of the instructions, (2) License file to clearly indicate under what license the software is released.

Some minor comments:
C3-9) 1. Figures 2 to 5 show the same plots for 4 different job types. I don’t think this is necessary and some space could be saved.

Response 3-9) Thank you for your feedback on figures 2 to 5. We can see why you would make that suggestion since the visual impression of figures 2 to 5 share visual similarities. However, we intended for all the main figures to clearly convey the key findings of our benchmark tests. That is, the saturation points of the scalabilities (where the cost curve

becomes a vertical line), the minimum achievable cost (where the cost curve becomes a horizontal line), and the optimal parallel computing settings are different among Refine3D, Class3D, Class2D, and Polish. We would appreciate it if you could consider keeping figures 2 to 5 the main figures, provided space permitting.

C3-10) 2. Figure 1a contains a typo - it should be UCSF Chimera(X).

Response 3-10) Thank you for finding a typo. It is fixed.

C3-11) In conclusion, although the work is technically correct, I don’t think there is much novelty in this article. The benchmarks and conclusions are specific to the two examples studied, and there have been other benchmarks published. If it is meant to be a practical solution, then there are many questions around how someone would deploy the GoToCloud system, and how users would use it for their datasets. I cannot recommend publication.

Response 3-11) We are grateful with your variable comments that allow us to remove the ambiguities of our writing, especially ones related the novelty of our work and our aim of each testing. As addressed in Response 3-2, the novelty of our current paper lies in the newly proposed architectural design of the GoToCloud platform. This design enables researchers to almost instantly access a powerful, ready-to-use computational platform for cryo-EM SPA. Users can easily utilize multiple AWS instance types (e.g., G5, G4dn, C6i) within a single Cryo-EM SPA workflow. This feature facilitates high efficiency for typical RELION workflows, which consist of multiple steps employing different algorithms. Each of these algorithms requires specific configurations of CPUs, GPUs, and memory for optimal performance. Additionally, the fast access speed of FSx for Lustre enhances the performance of all algorithms. Importantly, benchmark tests have led us to suggest optimal parallel computing settings, including the AWS EC2 instance type, for each of RELION's Refine3D, Class3D, Class2D, and Polish steps. The significant interest from over a dozen research groups who have requested the detailed results of our benchmark tests underscores the need for a comprehensive resource within the Cryo-EM SBDD community. To disseminate this valuable information and foster further research, we would like to share our findings in this publication.

The deployment of the GoToCloud platform and how users can create and use the platform instances for their datasets are addressed in Responses 1-1 and 3-8. Thus, it is possible for readers to build and deploy their own GoToCloud platform. To help such

readers, we added a README file in our GoToCloud GitHub. The file contains the link to the instruction of how to construct the shared EFS and S3, as well as how to install the relevant software, such as RELION. The README file also describes the usage of the GoToCloud scripts step-by-step to create and set up a GoToCloud platform instance on user's AWS account.
We hope our responses clarify all of your questions and concerns.

Version 1:

Reviewer comments:

Reviewer #1

(Remarks to the Author)
Thank you for addressing my comments.

Reviewer #2

(Remarks to the Author)
Brief Summary
The authors provided very informative and complete responses to my questions.

Overall Impression
The improvements from optimizing Relion with CUDA_ARCH=86 versus CUDA_ARCH=75 are interesting. While the improvements were not uniform across the different processing steps, the speedups would save on the computing costs. Moreover, compiling software to run optimally on the hardware is the kind of optimization that would be helpful for general users who might not be familiar with GPU hardware.

Specific Comments
I only have specific comments about wording.
Line 223 - "the ground total cost" probably should be "the grand total cost"
Line 308 - "Core GPUs were tempted" probably should be "Core GPUs were attempted"
Supplemental, Line 300 - "more pronoun as the number of nodes" probably should be "more pronounced as the number of node"

Reviewer #3

(Remarks to the Author)
My main concern about the original manuscript was whether their solution was of general interest, or whether it was just another local solution. An additional concern was that only limited tests were presented for the benchmarks, so that it was unclear whether their proposed optimisations were generally applicable.

The authors have clarified that the GTC scripts have been used by others for over a year, as well as at the Photon Factory, which gives me some reassurance.

I highlighted the lack of documentation on the Github site, and Reviewer 1 also asked for a README file and a licence file. There is now a README file with a link to more detailed documentation (I haven't tried to follow the steps myself, but they look ok). There is a GPLv3 license file, although I can't find any source code headers specifying which files are covered by the license.

The authors have spelled out their policy for updating software versions. I understand the time required to maintain compatibility, but it is important that GoToCloud doesn't become obsolete in a year's time. I can see instructions for installing Relion5 under https://sites.google.com/sbrc.jp/gotocloud-install/installation but it is not clear whether any of the GTC scripts have been updated.

I am less convinced by the response on variation in datasets. The rebuttal states "users are not expected to conduct

benchmark tests to find the optimal parallel settings for each of their datasets". I believe the implication is that users should use the optimum setup from these benchmarks (Figs 2 – 5) for a given job type. So the question is, are these results transferable to other datasets? The authors argue that the relative performance of different compute configurations is constant, even if the absolute timings and costs change. This is just an assertion. As far as I can see, it has not been tested to see if it is true.

Anecdotally, Relion is not considered to scale well to multiple nodes, and probably most people use a single multi-GPU node. The vertical sections of the plots in Figs 2 – 5 are certainly believable. But the fine detail of whether to use 16@2x g4dn(8,96) or 8@1x g5(8,192) or 8@2x g5(4,48) for Refine3D seems a lot less certain. The Discussion again states "it is important to use the optimal settings for each combination of computer hardware and image processing job". If it is important, then I don't want to rely on benchmarks from a different system.

So a follow up question is what freedom does a user have? IIUC step 2 will create a pcluster with pre-defined EC2 instance types. Can users still select within these instance types and choose the number of nodes, threads, etc from within the Relion GUI accessed via NICE-DCV? In other words, are the benchmarks advice, or does GTC lock the user in?

In conclusion, I am mostly happy with the revised manuscript and it is suitable for publication. The extra detail is helpful, and the online documentation looks good. For the benchmarking, ideally I would have liked to see a second example from a different system (different resolution, different symmetry, multiple 3D classes, etc). If that is not possible, then the conclusions should be toned down. I am not convinced by the "rationale" in Response 3-6 and I don't think the additional text should be added. It would be better to simply say that the benchmarking is an example only.

With that one change, I can recommend publication.

Author Rebuttal letter:

1

2 Response to Reviewers
3
4 Responses to Reviewer #1 (Remarks to the Author):
5 C1-1) Thank you for addressing my comments.
6
7 Response 1-1) Thank you again for taking the time to review our manuscript. Your
8 feedback was very helpful in improving the accessibility and usability of our software.
9
10
11 Responses to Reviewer #2
12
13 Brief Summary
14 The authors provided very informative and complete responses to my questions.
15
16 Thank you for your positive feedback. We are pleased that our responses adequately
17 addressed your questions. Your expert comments on AWS were valuable in helping us
18 improve the quality of our paper.
19
20 Overall Impression
21 The improvements from optimizing Relion with CUDA_ARCH=86 versus
22 CUDA_ARCH=75 are interesting. While the improvements were not uniform across the
23 different processing steps, the speedups would save on the computing costs. Moreover,
24 compiling software to run optimally on the hardware is the kind of optimization that
25 would be helpful for general users who might not be familiar with GPU hardware.
26
27 We are grateful for your suggestion to optimize RELION with CUDA_ARCH=86 versus
28 CUDA_ARCH=75, which led to further improvements in processing speed and cost
29 performance.
30
31 Specific Comments:
32 C2-1) I only have specific comments about wording.
33 Line 223 - "the ground total cost" probably should be "the grand total cost"
34 Line 308 - "Core GPUs were tempted" probably should be "Core GPUs were attempted"

1
35 Supplemental, Line 300 - "more pronoun as the number of nodes" probably should be
36 "more pronounced as the number of node"

Response 2-1) Thank you for identifying the grammatical mistakes. All three mistakes above have been corrected as you suggested.

Responses to Reviewer #3

C3-1) My main concern about the original manuscript was whether their solution was of general interest, or whether it was just another local solution. An additional concern was that only limited tests were presented for the benchmarks, so that it was unclear whether their proposed optimisations were generally applicable.

The authors have clarified that the GTC scripts have been used by others for over a year, as well as at the Photon Factory, which gives me some reassurance.

Response 3-1) We are glad to hear that the clarification regarding the use of the GoToCloud platform and scripts by others provided you some reassurance. We hope that our current paper conveys our message that our approach can be applied practically beyond our immediate research environment, supporting its relevance and general interest. Thank you for acknowledging this aspect of our work.

C3-2) I highlighted the lack of documentation on the Github site, and Reviewer 1 also asked for a README file and a licence file. There is now a README file with a link to more detailed documentation (I haven't tried to follow the steps myself, but they look ok). There is a GPLv3 license file, although I can't find any source code headers specifying which files are covered by the license.

Response 3-2) Thank you for taking the time to review our GitHub site as well. Following your suggestion, we have added a header statement about the license and copyright to all the source code files covered by the license.

C3-3) The authors have spelled out their policy for updating software versions. I understand the time required to maintain compatibility, but it is important that GoToCloud doesn't become obsolete in a year's time. I can see instructions for installing

Relion5 under https://sites.google.com/sbrc.jp/gotocloud-install/installation but it is not clear whether any of the GTC scripts have been updated.

Response 3-3) We are grateful for your taking the time to review our online documentation as well. To address your concern about an update of the GTC scripts in conjunction with the installation of RELION5-beta, we designed the platform architecture and GTC scripts to ensure that the scripts do not need to be updated whenever a new version of RELION or other software is installed. As explained in the manuscript and on the online instruction page, all supported versions of RELION and their associated "modulefiles" are pre-installed and created on the shared EFS. The default version is specified through the environment modules. We hope that the explanation clarifies your concern.

C3-4) I am less convinced by the response on variation in datasets. The rebuttal states "users are not expected to conduct benchmark tests to find the optimal parallel settings for each of their datasets". I believe the implication is that users should use the optimum setup from these benchmarks (Figs 2 – 5) for a given job type. So the question is, are these results transferable to other datasets? The authors argue that the relative performance of different compute configurations is constant, even if the absolute timings and costs change. This is just an assertion. As far as I can see, it has not been tested to see if it is true.

Response 3-4) Thank you for your valuable comments. While we agree that our rebuttal statement is a logical deduction that needs to be tested, repeating benchmarks with additional datasets was not feasible within the scope of this revision. Therefore, we revised the additional text for Response 3-6 in our previous rebuttal to clarify that our results are an illustrative example rather than a comprehensive validation. The revise also acknowledges the importance of avoiding overly generalized conclusions based on a single example. At the same time, we also address the C3-5 and C3-7 with this revised paragraph (please refer Response 3-5 and Response 3-7). Accordingly, we modified the corresponding paragraph in lines 382-398 of the main text, stressing in the last sentence that the transferability of these results to other datasets needs to be tested in future work

(see below).

p. 17-18, lines 382-398.

3

“An important consideration is the general applicability of the proposed optimal parallel computing settings for each RELION job type and/or other datasets. Our benchmarking results, based on a single system and dataset, serve as an illustrative example specific to the configuration used in this study. While these results offer valuable insights into optimizing parallel computing settings on the GoToCloud platform, they are primarily intended to help users customize these settings according to their research priorities, as discussed above. Unlike on-premise computing resources, which are often not publicly accessible and likely offer limited availability to external users, the AWS cloud computing service is publicly available. This accessibility makes it easier for researchers in both academia and industry to optimize parallel computing settings on the same platform according to their needs by referring our benchmark results. Nevertheless, our benchmarking results should be viewed as an example rather than a comprehensive validation. Therefore, outside the GoToCloud platform, we encourage readers to conduct their own benchmarks on their specific systems, to achieve optimal performance. Additionally, the transferability of these results to other datasets is another important consideration, and this should be explored in future work to fully automate the cryo-EM SPA workflow.”

C3-5) Anecdotally, Relion is not considered to scale well to multiple nodes, and probably most people use a single multi-GPU node. The vertical sections of the plots in Figs 2 – 5 are certainly believable. But the fine detail of whether to use 16@2x g4dn(8,96) or 8@1x g5(8,192) or 8@2x g5(4,48) for Refine3D seems a lot less certain. The Discussion again states “it is important to use the optimal settings for each combination of computer hardware and image processing job”. If it is important, then I don’t want to rely on benchmarks from a different system.

Response 3-5) Thank you for pointing out the ambiguity regarding the important aspect of the applicable range of our benchmark tests. We agree that users should not rely solely on benchmarks from a different system but should consider them as a guideline or starting point for finding optimal parallel computing settings on their own hardware. Therefore, we clarified that our results are an illustrative example rather than a comprehensive validation and overly generalized conclusions based on this single example should be avoided. This point is also addressed in Response 3-4.

C3-6) So a follow up question is what freedom does a user have? IIUC step 2 will create a pcluster with pre-defined EC2 instance types. Can users still select within these instance

4

types and choose the number of nodes, threads, etc from within the Relion GUI accessed via NICE-DCV? In other words, are the benchmarks advice, or does GTC lock the user in?

Response 3-6) Thank you for your interest in the GoToCloud platform. Through the RELION GUI application in the NICE-DCV WEB page of GoToCloud, users can choose a Slurm partition, which is associated with a specific EC instance type, for each RELION job from a set of available partitions and specify the number of nodes and threads. To depict this, we added Supplemental Fig. S2b that shows a terminal window with the list of available partitions printed by the Slurm “sinfo” command. Finally, this is why we provide the details of the benchmark results as guidance or a starting point of the EC instance type selection for users.

C3-7) In conclusion, I am mostly happy with the revised manuscript and it is suitable for publication. The extra detail is helpful, and the online documentation looks good. For the benchmarking, ideally I would have liked to see a second example from a different system (different resolution, different symmetry, multiple 3D classes, etc). If that is not possible, then the conclusions should be toned down. I am not convinced by the “rationale” in Response 3-6 and I don’t think the additional text should be added. It would be better to simply say that the benchmarking is an example only.

With that one change, I can recommend publication.

Response 3-7) Thank you for your thorough review and for your positive feedback on the revised manuscript and the online documentation. Regarding your feedback on Response 3-6 of our previous rebuttal, we appreciate your input and agree that the additional text should be removed. We modified the text added in Response 3-6 of our previous rebuttal based on your suggestion. This point is also addressed in Response 3-4.

We hope these changes address your concerns, and we appreciate your constructive suggestions.