Supplementary Information

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

Toshio Moriya[1,†,*], Yusuke Yamada[1,2,†], Misato Yamamoto[1], and Toshiya Senda[1,2,*].

[1]*Structural Biology Research Center, Photon Factory, Institute of Materials Structure Science, High Energy Accelerator Research Organization (KEK), 1-1 Oho, Tsukuba, Ibaraki 305-0801, Japan.*

[2]*Department of Materials Structure Science, School of High Energy Accelerator Science, The Graduate University of Advanced Studies (Soken-dai), 1-1 Oho, Tsukuba, Ibaraki 305-0801, Japan.*

[†]These authors contributed equally.

*These authors jointly supervised this work. Correspondence and requests for materials should be addressed to T.M. (email: toshio.moriya@kek.jp) and T.S. (email: toshiya.senda@kek.jp).

Table of Contents

1. Supplementary Methods

1.1. GoToCloud system mapping

As shown in **Fig. 1b,** GoToCloud ensures security of the analysis environment on a per-research-group basis by utilizing the robust security services provided by default in AWS. A logically isolated virtual network dedicated to each AWS account can be easily established using the Amazon Virtual Private Cloud (VPC) service. Additionally, AWS storage services, such as the Amazon Simple Storage Service (S3), Amazon Elastic File System (EFS), and Amazon FSx for Lustre (Lustre), do not allow access from the other AWS accounts without explicit settings, making it easy to guarantee security. By explicitly allowing access to analysis data only among these isolated virtual networks, access to the analysis data by other research groups can be prevented.

Each user of a research group is linked to an AWS Identity and Access Management (IAM) user. Within an AWS account owned by a research group, the access rights for each IAM user to AWS services are created and set by the AWS account administrator of the respective group based on the roles assigned to the user. Currently, the default setting of GoToCloud allows each user to freely create, edit, and delete necessary services for the creations and deletions of virtual cluster instances through the "AWS ParallelCluster (pcluster)" service. Nevertheless, it is also possible to restrict user access rights to allow only use of existing virtual cluster instances without allowing creation or deletion of instances.

Each project is associated with a Cloud-based integrated development environment (IDE) called AWS Cloud9 (Cloud9). By using Cloud9, management of the virtual cluster service, such as creation and deletion of pcluster instances, can be easily performed. Though it is often crucial to assign access restrictions dedicated to a specific user for a

particular analysis project, there are cases where it is advantageous for multiple users to share their analysis works with others. This is particularly beneficial when multiple users collaborate on the same analysis project as a team. In this case, the project owner can easily handle this use case by registering the IAM users of other team members on the shared list of Cloud9 instances for the project.

The information about an analysis environment is mapped to a GoToCloud configuration (GTC config). Each of the different analysis methods requires a unique configuration of the analysis environment, including the hardware specifications of the virtual cluster and the set of required software packages optimal for a given analysis method. A GTC config contains the descriptions of how to set up the hardware and software for a given analysis method on a virtual cluster instance upon the creation. The details of GTC config are described in the next section, **section 1.2**.

1.2. Multi-account and multi-region support of the GoToCloud shared EFS

The algorithms for Cryo-EM data processing are evolving daily now; new algorithms are proposed regularly, and existing software packages have been frequently updated. To use the latest analysis software packages in a sustainable fashion, users need to add new packages and reinstall each of the existing ones whenever an update becomes available. However, selecting which software package should be included in the computational platform for a given analysis method requires specialized knowledge, and one must always keep track of the updates, such as version upgrades or patch releases. Worse yet, there are multiple software packages. Therefore, it is an urgent issue to have a system that can quickly reflect these updates to the analysis platform.

GoToCloud addressed these issues by providing a shared EFS where a set of software packages required for a given analysis method has already been pre-installed by the

GoToCloud management group who manages and maintains the system (**Fig. 1b and Supplementary Fig. S7**). The shared EFS has an additional benefit of reducing the time required to set up the software environment on the head node of a pcluster instance right after the instance is created by the Step 2 GTC script (gtc_pcluster_create.sh) (**Supplementary Fig. S1**; see also **section 1.4**). This script also installs the minimum number of dependent software components which are required by these analysis software packages and must be installed on the OS, such as shared libraries and drivers. It essentially realizes GTC config by generating a "config.yaml" file for the creation of a pcluster instance from templates prepared by the GoToCloud management group. The generated config.yaml file contains the uniform resource locator (URL) for the "post_install.sh" shell script that is stored in S3 bucket owned by the GoToCloud management group, and the reference to the shared EFS is pre-defined in the post_install.sh script.

This GoToCloud architecture allows experts to accumulate their knowledge and experience of software environment setup in the shared EFS. Since all AWS accounts can share the same EFS in GoToCloud, all users can immediately use the latest analysis algorithms reflected in an optimal setup of the software environment. This way, all GoToCloud users are freed from the nuisance requirement of keeping up the software updates by themselves. Thus, even research groups that do not have sufficient knowledge and experience in designing and constructing an analysis software environment can immediately use the latest one prepared by the experts.

While sharing the analysis software environment among multiple accounts, it is still crucial to ensure robust security at the AWS account level. The AWS VPC peering (Peering) service can establish a secure connection between two VPCs even if they are in

two different AWS accounts. To establish a Peering connection between the VPC for the GoToCloud shared EFS and a VPC in the user's AWS account (**Supplementary Fig. S7**), the GoToCloud management group needs to do the following. (1) Ask the user for the AWS Region and AWS account number. (2) Add the user-provided information along with the IP address of the VPC associated with the shared EFS (i.e., the configuration of the Peering connection) to the AWS CloudFormation template (a YAML formatted text file). (3) Finally, provide the template back to the user. The AWS CloudFormation template has descriptions of all the AWS resources and their properties necessary to create the GoToCloud platform on the user's account as a stack of the AWS CloudFormation. By using this AWS CloudFormation template for the initial setup of the GoToCloud platform in the user's AWS account, users can easily set up a Peering connection as well as the Cloud resources for GoToCloud by themselves.

However, there is one practical issue: AWS currently allows only up to 125 Peering connections per VPC. Furthermore, only Peering connections between two VPCs that exist within the same AWS Region (a unit of AWS's cloud server data center group) are free of charge. There are numerous numbers of AWS Regions, which are geographically separated from each other. In GoToCloud, a mirror EFS of the shared EFS, which is the master in the GoToCloud management group account, is prepared for each AWS Region for the Peering to overcome the connection limit and provide a cost-efficient multi-region support (**Supplementary Fig. S7**).

1.3. Implementation of the Cryo-EM SPA software environment

In GoToCloud, Slurm is adapted as the job scheduler for pcluster, and RELION[1,2] is also set up to use Slurm. This allows the actual SPA calculations to be performed on the virtual compute nodes (or simply "nodes") of a pcluster instance. We selected m5.xlarge

for the head node, which is not expected to perform most of the data processing. Instead, it was set up as a machine with sufficient specifications for displaying Cryo-EM maps using UCSF Chimera[3] on a desktop environment connected via NICE DCV (**Supplementary Fig. S2**).

1.4. GoToCloud (GTC) scripts

The initial setup of GoToCloud in a user's AWS account is performed with a template of AWS CloudFormation which is prepared by the GoToCloud management group, as described in **section 1.2** above. This setup is necessary only once per AWS Region and per AWS account. The template includes the creation of a default IAM user and a VPC instance. It configures the default IAM user to have sufficient permissions for the creation and deletion of pcluster instances, and the VPC to have an access permission to the shared EFS. This task is performed by the administrator of a user's AWS account. After the setup, the administrator can add IAM users using the default IAM user group configuration as a reference.

To use the GoToCloud (GTC) scripts, the user must first login to the AWS account of the user's own research group with the user's own IAM user from the AWS Management Console; then, the user must create a Cloud9 environment for a project. Here, only two tasks are required of the user: (1) providing a project name and (2) selecting the user's predefined VPC for Peering connection to the VPC of the shared EFS. Importantly, the project name must comply with the AWS requirements of non-US standard regions. This is because the GTC scripts newly create a S3 bucket as well as an EC2 key pair (key pair) and at least one pcluster instance for each project, and the naming of each S3 bucket must be globally unique and fulfil the above requirements. To significantly reduce the number of the input parameters required for the user with the GTC scripts, a naming format "[IAM

username]-[AWS account ID]-[project name]" specific to the GoToCloud platform is used as a unified name of all the above service instances in each project.

Once the creation of the Cloud9 environment is completed, the user opens the terminal in the Cloud9 environment to execute the GTC scripts one by one (**Supplementary Fig. S1**). The Step 1 GTC script (gtc_setup_gotocloud_environment.sh) prepares the creation of pcluster instances. This script performs the following 8 setups altogether: (1) mounting the shared EFS, (2) installing the pcluster library, (3) installing dependencies (e.g., jq and Node.js), (4) checking for the existence of the necessary role for managing spot instances startup (i.e., EC2Spot Role), and creating it if it does not exist, (5) setting the Cloud9 tags for cost management (see **Supplementary Table S4**), (6) creating an S3 bucket for uploading the input cryo-EM dataset and long-term storage for the analysis outputs, (7) generating an EC2 key pair for SSH connection to the head nodes of pcluster instances, and (8) creating the pcluster configuration file ("config.yaml") tailored to a default analysis method (Cryo-EM SPA with RELION4.0.1). Currently, the Linux distribution of this plucster instance is set to Ubuntu 20.04 LTS (Long Term Support) in the pcluster configuration file. For the standard usage, there are no parameters that the user must specify in this script. A project specific S3 bucket is created in this step, and the input dataset (e.g., a set of micrograph movie files) can be uploaded to this bucket. As examples, the averages of times to transfer a 1.0 terabyte (TB) input dataset (EMPIAR-1058) for three times (i.e., n=3 independent data transfers) 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).

The Step 2 GTC script (gtc_pcluster_create.sh) creates a new pcluster instance for the project. This process takes about 15–30 minutes. As in the case of the previous script, the user does not need to specify any parameters under normal usages. Once the instance is created, the contents of the S3 bucket associated with Lustre mounted on this pcluster instance are automatically synchronized. The script also automatically mounts the shared EFS on the head node, and then sets up the system environment for pre-installed software packages in the shared EFS.

The final Step 3 GTC script (gtc_pcluster_dcv_connect.sh) obtains the URL of the NICE-DCV remote desktop environment on the head node of the pcluster instance created in Step 2 (**Supplementary Fig. S2**). Additionally, it is possible for the user to connect to the head node via SSH using another GTC script (gtc_pcluster_ssh.sh). Currently, GUI applications cannot be used with SSH connections since X-Forwarding is not supported by the Cloud9 environment.

When the user wishes to pause analysis work and does not plan to use this GoToCloud platform instance for a while, it is recommended that the user delete the pcluster instance to stop billing for the relatively expensive Lustre and head node. The Step 4 GTC script (gtc_pcluster_delete.sh) is prepared for this purpose. It automatically exports the data of the Lustre to the S3 bucket before deleting the pcluster instance. The deletion process usually takes about 15–30 minutes depending on the amount of data that needs to be exported. Again, this script does not require any parameters to be specified by the user.

To resume the analysis work, the user goes back to Step 2. By repeating the steps from Step 2 to Step 4 during a project, the user can reduce the total cost for the project.

When the user has completed the project and no longer needs to use the GoToCloud platform instance, the user can use the Step 5 GTC script (gtc_aws_delete_s3_and_key_pair.sh) to delete the associated S3 bucket and key pair. Upon execution, all the data stored in the S3 bucket is forcibly deleted, so the user must make sure to archive any necessary data for later usage to another storage location before executing this script. Finally, the user deletes the Cloud9 for the project from the AWS Management Console.

2. Supplementary Tables

Supplementary Table S1. Information of EMPIAR datasets

	Nitrite Reductase EMPIAR-10581 EMD-0731	**Streptavidin EMPIAR-10641 EMD-30913**
Data collection		
Microscope	Talos Arctica G2	Titan Krios G3i
Voltage (kV)	200	300
Detector	Falcon 3EC	K3 Bioquantum
Energy filter	-	GIF Bioquantum (15eV slit)
Electron dose ($e^-/Å^2$)	49.0	70.0
Pixel size (Å/pixel)	0.880	0.400
Estimated defocus range (μm)	- 0.5 to -3.0	-0.3 to -2.6
Dataset size	1.0 TB (MRC)	373.1 GB (TIFF)
Micrographs	694	2,277
Extracted particles	176,256	(Not reported)
Reconstruction		
Main software	RELION4.0	RELION3.1
Final particles	89,513	19,045
Theoretical weight (kDa)	110	75
Symmetry	C_3	D_2
EMD map resolution (Å)	2.85	1.93

Supplementary Table S2. Specification of AWS EC2 instance types used for NiR benchmark tests

EC2 Instance Type	GPUs	GPU Memory (GiB)	vCPUs	Memory (GiB)	Instance Storage (GB)	Network Band-width (Gbps)	EBS Band-width (Gbps)	On-Demand Price/Hour** (USD)
G5: NVIDIA A10G Tensor Core GPU, AMD EPYC 7R32 CPU (2nd generation AMD EPYC processors)								
g5.48xlarge	8	192	192	768	2x3800*	100	19	$16.288
g5.12xlarge	4	96	48	192	1x3800*	40	16	$5.672
g5.4xlarge	1	24	16	64	1x600*	~ 25	8	$1.624
g5.xlarge	1	24	4	16	1x250*	~ 10	~ 3.5	$1.006
G4dn: NVIDIA T4 Tensor Core GPU, Intel Cascade Lake P-8259L CPU (2nd Generation Intel Xeon Scalable processors)								
g4dn.metal	8	128	96	384	2x900*	100	19	$7.824
g4dn.12xlarge	4	64	48	192	1x900*	50	9.5	$3.912
g4dn.4xlarge	1	16	16	64	1x225*	~ 25	4.75	$1.204
g4dn.2xlarge	1	16	8	32	1x225*	~ 25	~ 3.5	$0.752
C6i: Intel Ice Lake 8375C CPU (3rd Generation Intel Xeon Scalable processors, 3.5GHz)								
c6i.32xlarge	N/A	N/A	128	256	EBS	50	40	$5.440

* NVMe SSD (Non-Volatile Memory express Solid-State Drive).

** The listed prices of EC2 instance types are for the us-east-1 (Northern Virginia) AWS Region. Note that these prices vary depending on AWS Region and may also vary depending on the period of use.

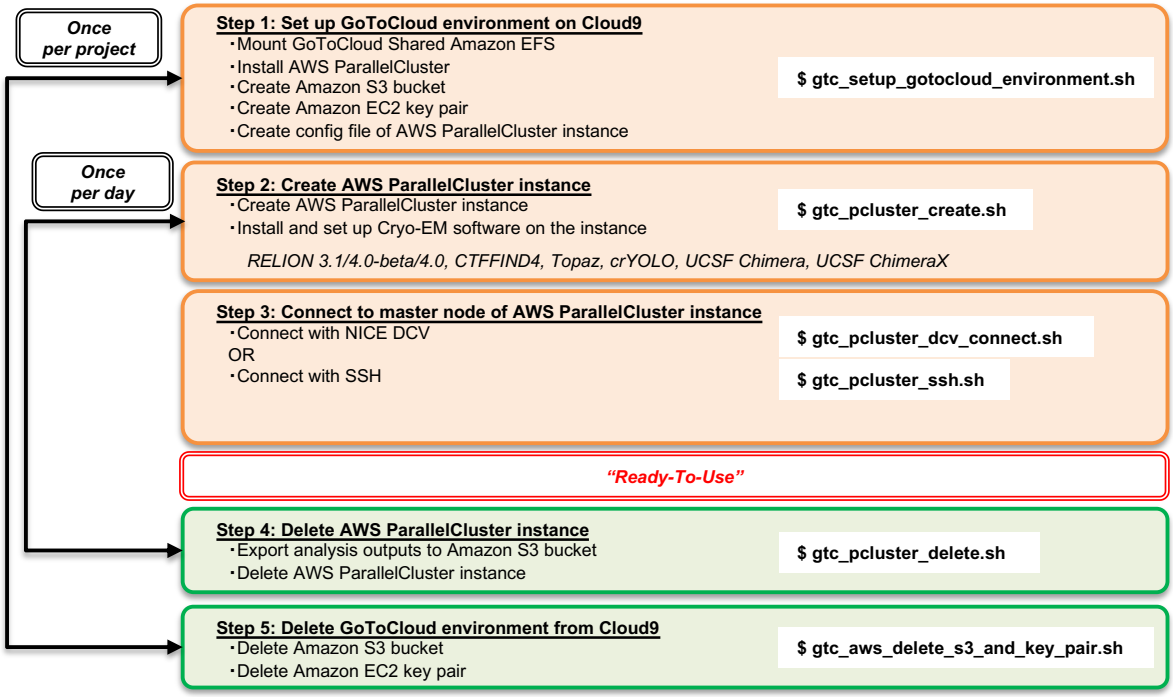
Supplementary Table S3. Input parameters of RELION jobs in the NiR benchmarks

	Refine3D	Class3D	Class2D	Polish
Rescaled box size [pixels]	352	64	64	352
Rescaled pixel size [Å/pixel]	1.17	3.3	3.3	1.17
Original box size [pixels]	468	240	240	468
Original pixel size [Å/pixel]	0.88	0.88	0.88	0.88
Mask diameter [Å]	164	120	120	-
Particles	129,298	129,298	176,256	129,298
Symmetry	C3	C1	-	-
Classes	-	4	200	-
Use fast subset	-	No	No	-

Supplementary Table S4. AWS service tags in the GoToCloud platform

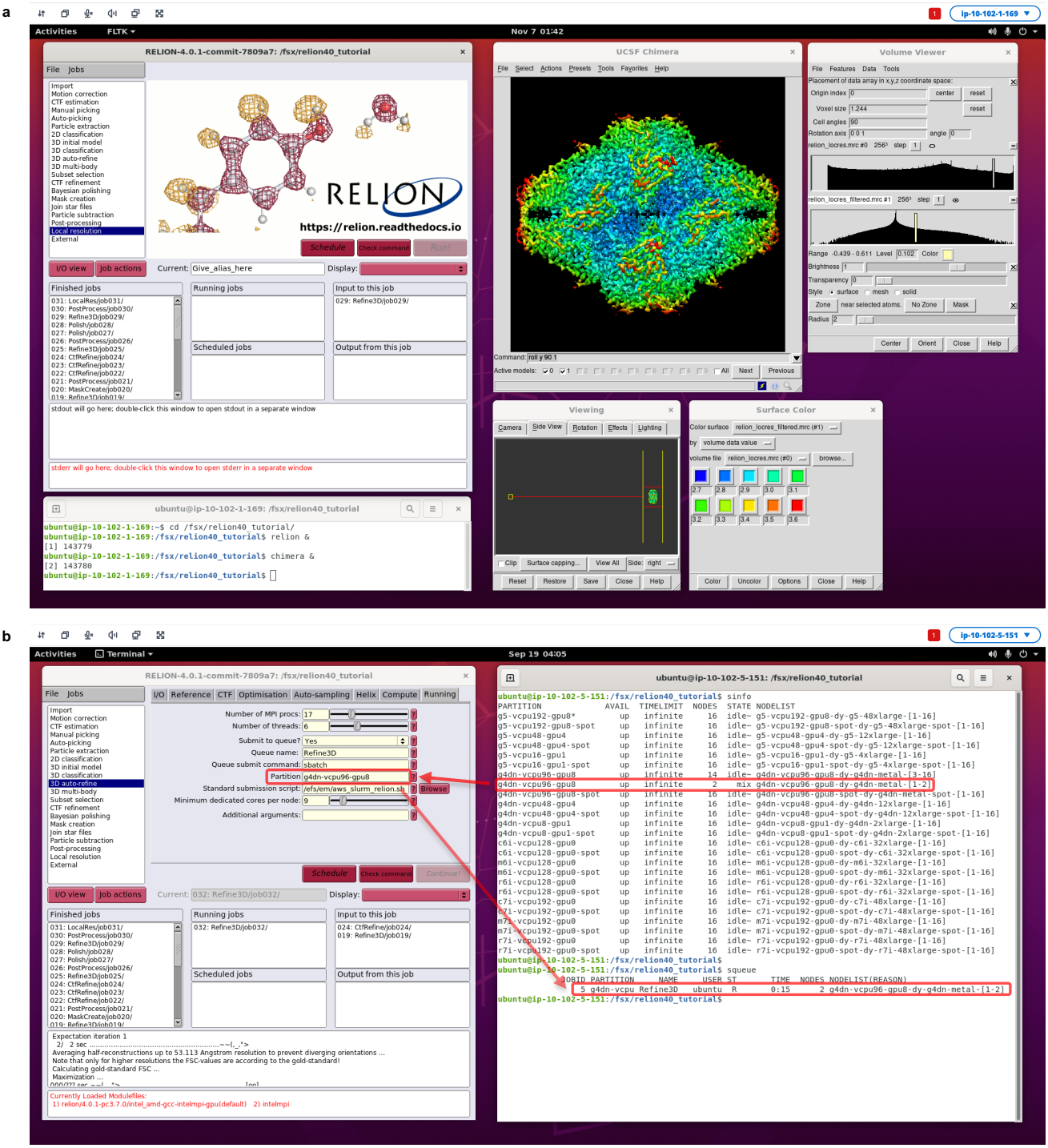
Tag Key	Tag Value	Tag Value Examples
gtc:account	AWS account number	327610663674
gtc:iam-user	IAM user ID	kek-gtc-user01
gtc:project	Project name	empiar-10581
gtc:method	Method name	cryoem

3. Supplementary Figures



Supplementary Figure S1. GoToCloud (GTC) scripts

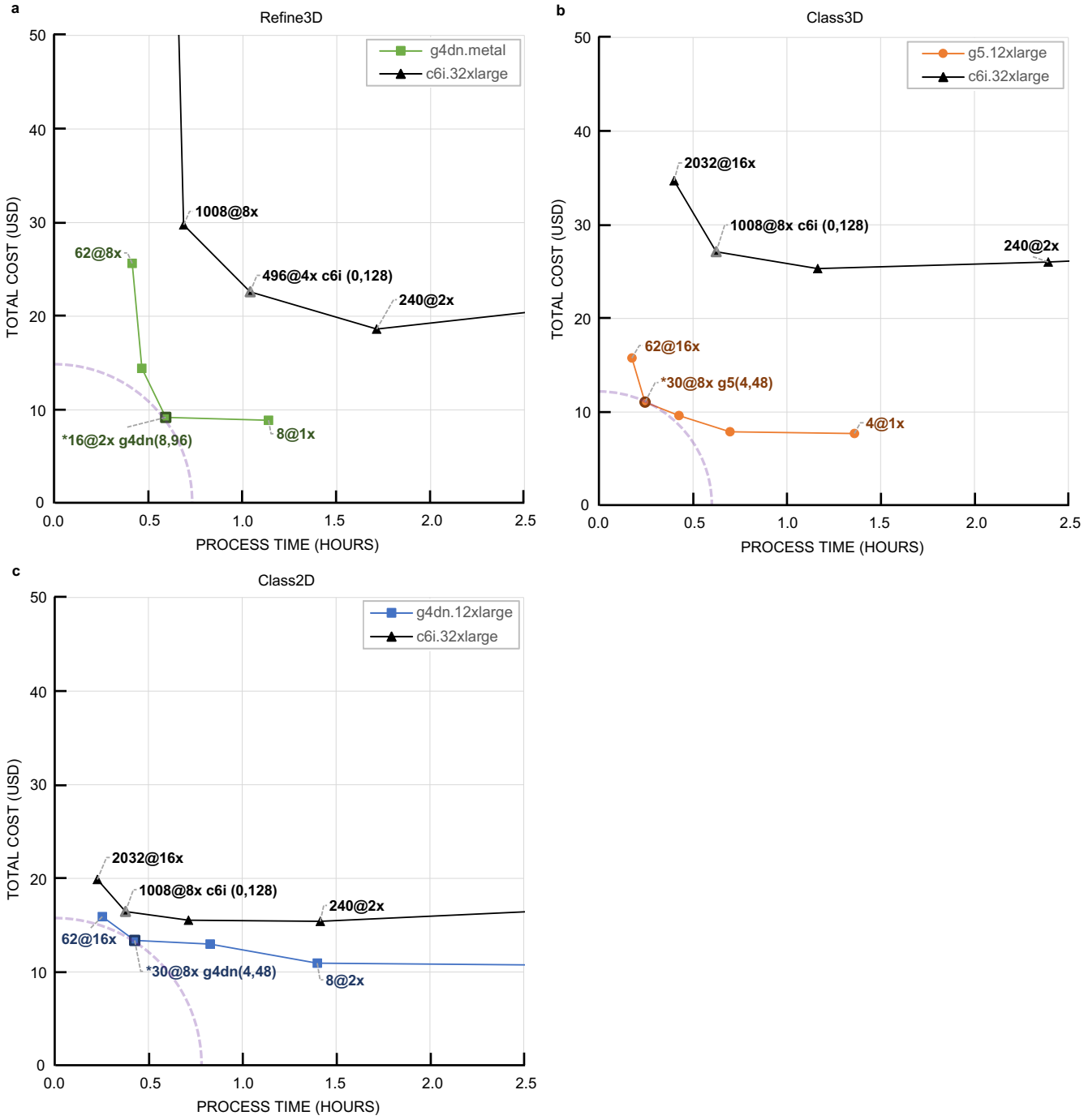
A set of scripts allowing the user to construct a ready-to-use Cryo-EM SPA computing platform in just three steps with GoToCloud. Steps 1 to 3 require approximately 30 to 60 minutes.



242 **Supplementary Figure S2. NICE-DCV screen**

243 User interface of GoToCloud. **(a)** GUI windows of RELION and UCSF Chimera. **(b)**

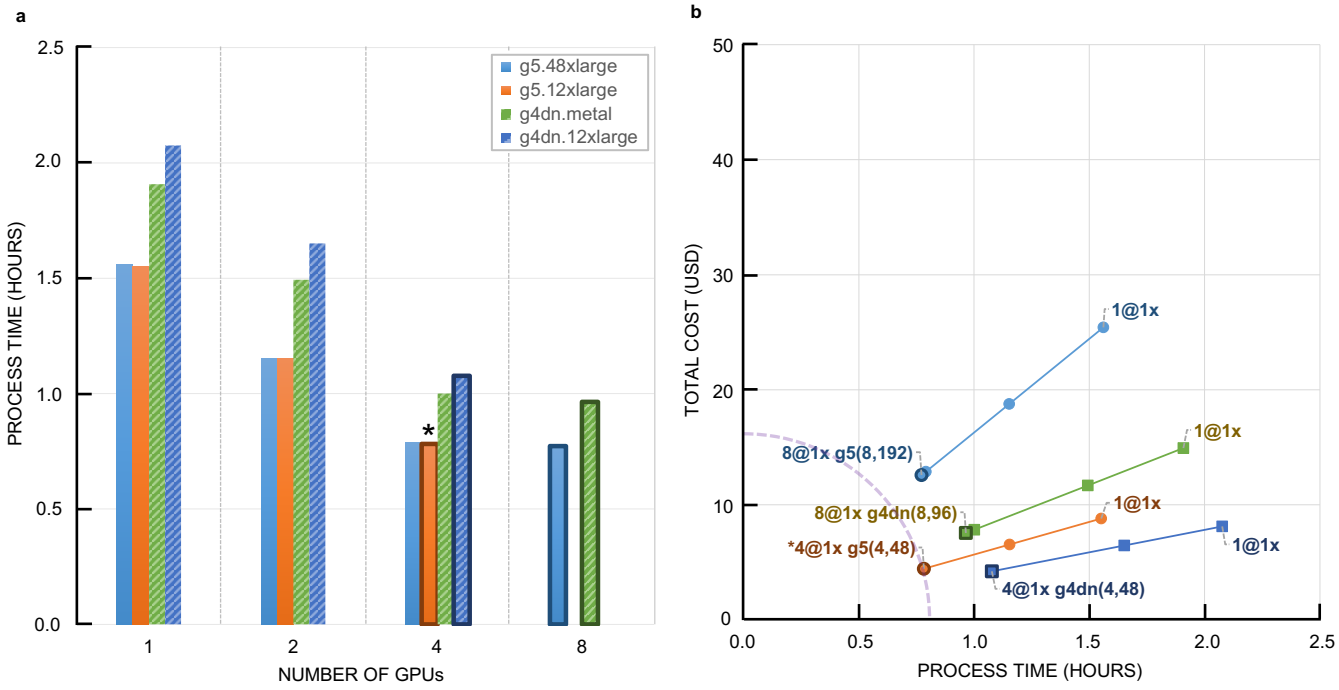
244 Selection of a Slum partition through RELION GUI application.



Supplementary Figure S3. CPU- vs GPU-optimized executions in NiR dataset benchmarks

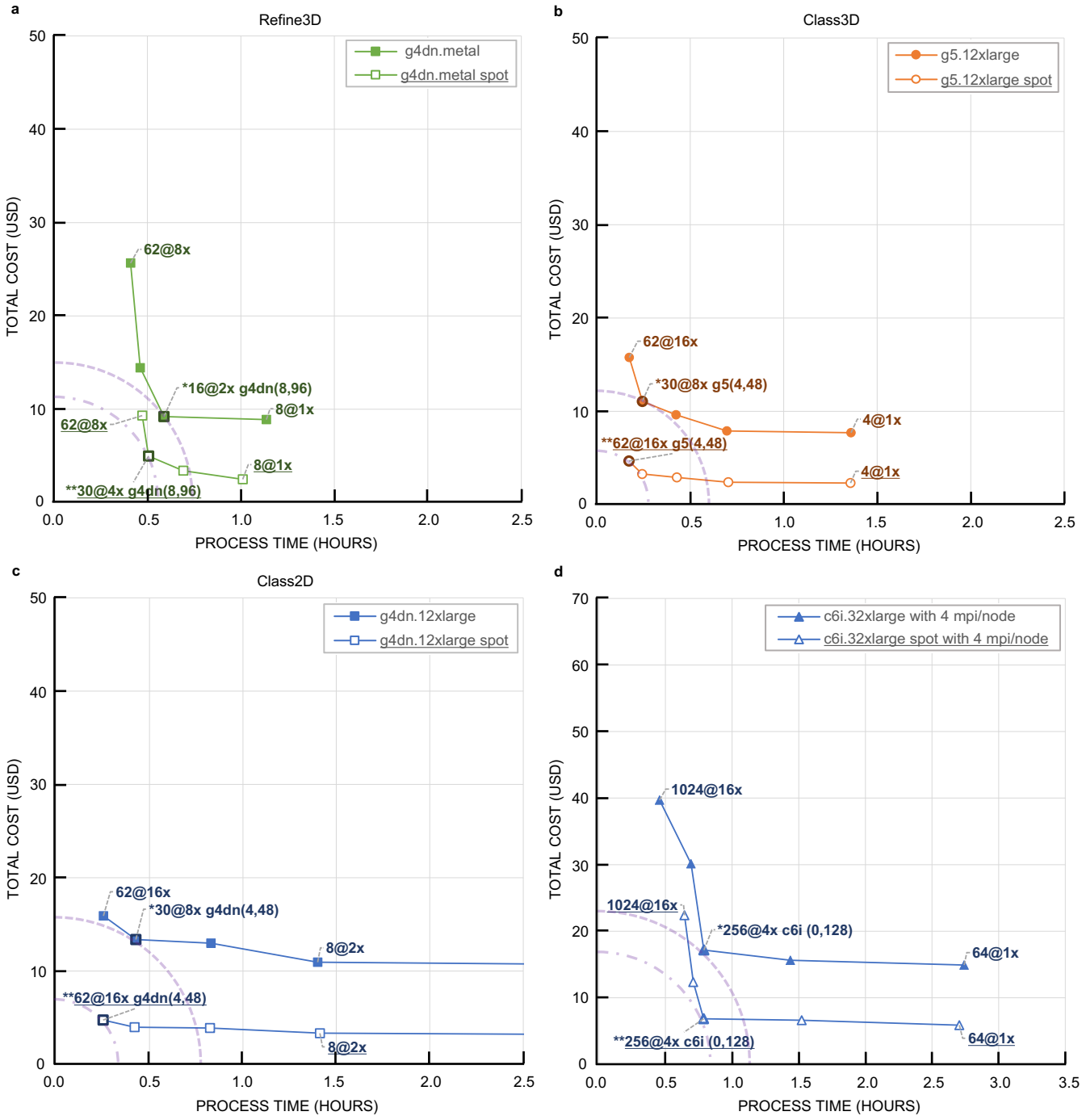
Scatter plots comparing the cost performance of CPU- vs GPU-optimized executions of **(a)** Refine3D, **(b)** Class3D, and **(c)** Class2D in the NiR dataset benchmarks. For CPU-optimized executions, the RELION executable **EXE02_CPU** in **Supplementary Data 1**

and c6i.32xlarge (black) from the C6i instance type group (triangles) were used. For comparison, the results of the AWS EC2 instance type including the configuration that achieved the globally optimal cost performance for each RELION job type in **Fig. 2b, 3b, and 4b** are also shown as the representatives of GPU-optimized executions. The GPU-optimized executions showed clearly better cost performances with all the RELION job types, but the difference in the Class2D benchmark was much smaller compared with the other jobs. In particular, the comparison between the optimal settings of the two execution types (outlined) shows that the process time and the total cost of the CPU-optimized execution relative to the GPU-optimized one increased approximately 1.76 and 2.45 times for Refine3D, and 2.56 and 2.45 times for Class3D, respectively. On the other hand, with Class2D, the process time was slightly shorter (0.88 times), but the total cost was higher (1.23 times). The axes, colors, marker symbols, and marker labels of GPU-optimized executions are the same as in **Fig. 2b, 3b, and 4b**. The marker labels of CPU-optimized executions are the same as in **Fig. 5b**. The raw data are provided in **Supplementary Data 2**.



Supplementary Figure S4. NiR dataset benchmark result of Class2D with the VDAM algorithm

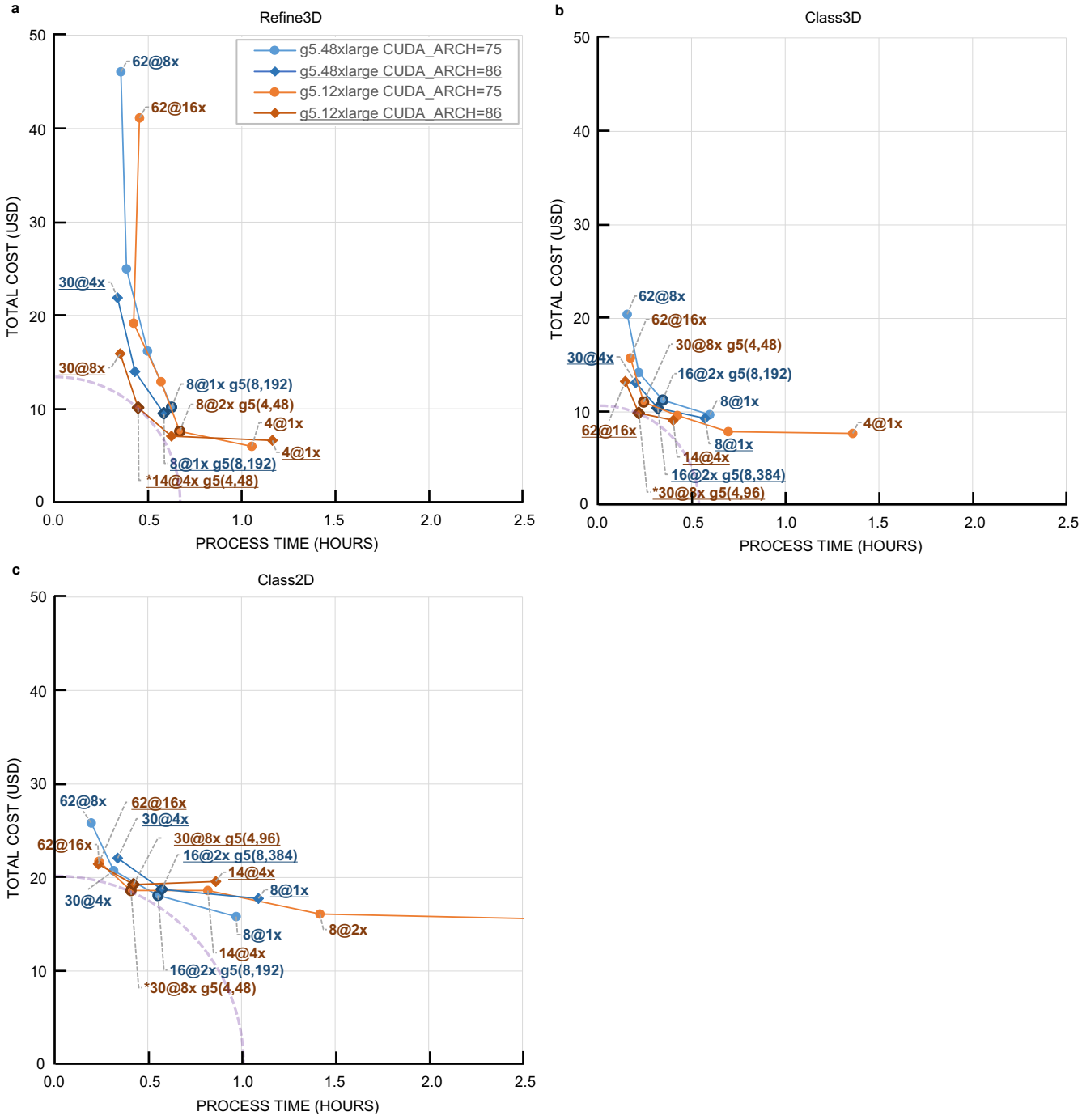
The bar graph and scatter plot of the result of Class2D with the VDAM algorithm. The same RELION executable and AWS EC2 instance types were used as in **Fig. 2**. Since the VDAM algorithm does not support MPI, the effect of the number of GPU cards used within a single EC2 instance type was investigated here. **(a)** Bar graph demonstrating that the scalability was saturated at 4 GPUs settings. **(b)** Scatter plot showing that the configuration of 4 GPUs settings with g5.12xlarge achieved the best cost performance. Interestingly, not only the process time but also the total cost of the 8 GPUs settings were almost identical to those of 4 GPUs with both g5.48xlarge and g4d.metal. The axes, colors, bars, marker symbols, and marker labels are the same as in **Fig. 2**. The raw data are provided in the **Class2D_VDAM** sheet of **Supplementary Data 2**.



Supplementary Figure S5 Spot instance vs on-demand executions in NiR dataset benchmarks

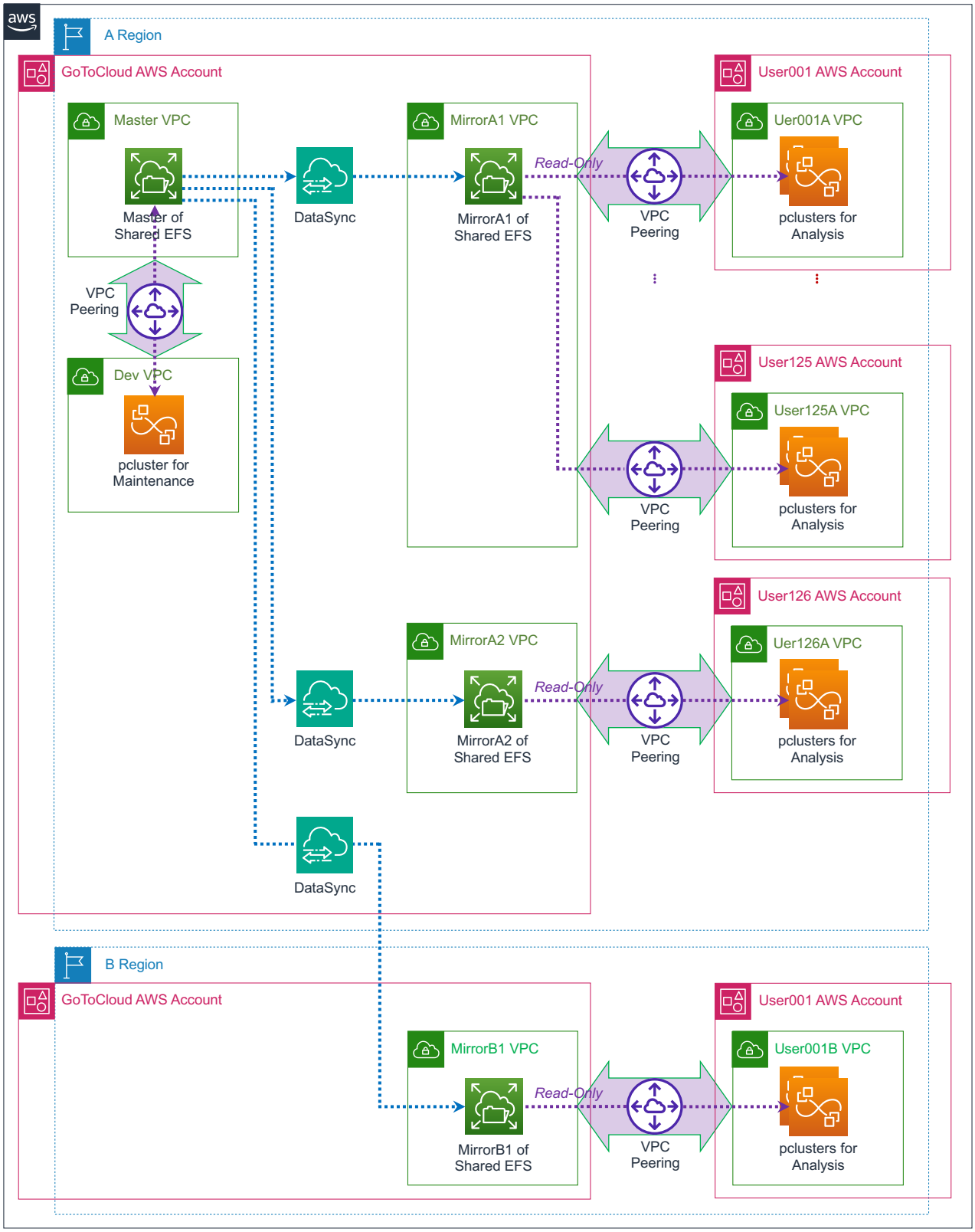
The Scatter plots comparing the cost performances between spot instance executions (unfilled) and on-demand executions (filled) of **(a)** Refine3D, **(b)** Class3D, **(c)** Class2D, and **(d)** Polish in the NiR dataset benchmarks. For the spot instance executions, the AWS

EC2 instance types including the configuration that achieved the globally optimal cost performance for each RELION job type in **Fig. 2b, 3b, 4b, and 5b** were used. For comparison, the on-demand execution results of the same EC2 instance types from these figures are also shown. The best cost performances were all achieved by the configuration with spot instances: 4 nodes of g4dn.metal (30 GPUs total) for Refine3D, 16 nodes of g5.12xlarge (62 GPUs total) for Class3D, 16 nodes of g4dn.12xlarge (62 GPUs total) for Class2D, and 4 nodes of c6i.32xlarge with 4 MIPs/node settings (256 vCPUs total) for Polish. The double asterisks (**) indicate these optimal configurations, and their distances from the origin are indicated by the additional dashed-dotted arcs. Regardless of REION job types and the configurations, the process times of spot instance executions remained almost the same, but the costs became much lower relative to the costs of on-demand executions in general. As a result, the optimal configurations tended to shift toward a larger computer resource side (i.e., more GPUs or vCPUs) with spot instances. The axes, colors, marker symbols, and marker labels are the same as in **Fig. 2b, 3b, 4b, and 5b**. The raw data are provided in **Supplementary Data 2**.



Supplementary Figure S6. CUDA_ARCH=75 vs CUDA_ARCH=86 build executions Scatter plots comparing the cost performance of CUDA_ARCH=75 vs CUDA_ARCH=86 build executions of **(a)** Refine3D, **(b)** Class3D, and **(c)** Class2D in the NiR dataset benchmarks. For the CUDA_ARCH=86, the RELION executable **EXE04_GPU** in **Supplementary Data 1** and g5.48xlarge (deep blue rhomboid) and

g5.12xlarge (deep orange rhomboid) were used. For the CUDA_ARCH=75 build execution, the data are the same as the ones of g5.48xlarge and g5.12xlarge in **Fig. 2b, 3b, and 4b**. The CUDA_ARCH=86 build executions showed clearly better cost performances with Refine3D, slightly better with Class3D. The improvements become more pronounced as the number of nodes increases. On the other hand, cost performance is almost the same or slightly worse with the Class2D. The axes, colors, marker symbols, and marker labels are the same as in **Fig. 2b, 3b, and 4b**. The raw data are provided in **Supplementary Data 2**.


aws
A Region
GoToCloud AWS Account
Master VPC
Master of
Shared EFS
VPC
Peering
Dev VPC
pcluster for
Maintenance
DataSync
DataSync
DataSync
MirrorA1 VPC
MirrorA1 of
Shared EFS
MirrorA2 VPC
MirrorA2 of
Shared EFS
Read-Only
Read-Only
VPC
Peering
VPC
Peering
VPC
Peering
User001 AWS Account
Uer001A VPC
pclusters for
Analysis
User125 AWS Account
User125A VPC
pclusters for
Analysis
User126 AWS Account
Uer126A VPC
pclusters for
Analysis
B Region
GoToCloud AWS Account
MirrorB1 VPC
MirrorB1 of
Shared EFS
Read-Only
VPC
Peering
User001 AWS Account
User001B VPC
pclusters for
Analysis

310

Supplementary Figure S7. Multi-account and multi-region support of the GoToCloud shared EFS

A diagram describing the implementation of the multi-account and multi-region support of the GoToCloud shared EFS. The AWS account (red rectangle) of the GoToCloud management group ("GoToCloud AWS Account") at left top has a VPC (green rectangle, "Master VPC") for securing the master of the GoToCloud shared EFS ("Master of Shared EFS") in an AWS Region (blue dotted rectrangle, "A Region"). The contents of the master EFS are maintained from a dedicated pcluster by the management group; the pcluster is placed inside another VPC ("Dev VPC") in the same AWS Region, and the setup of this VPC is customized for the maintenace purpose. Any updates made in Master of Shared EFS are synchronized to a mirror EFS (e.g., "MirrorA1 of Shared EFS") through the AWS DataSync service (blue dotted arrows). The mirror EFS is placed within its own VPC (e.g., "MirrorA1 VPC"). In each of the user's AWS accounts at right (from "User001 to User125 AWS accounts"), a VPC (e.g., from "User001A to User125A VPCs") is created and connected to the mirror VPC by a VPC Peering connection (purple dotted arrows). Consequently, the entire contents in Master of Shared EFS are accessible from the environments of all users, realizing the multi-account support. As a workaround for the 125-connection-limits of the AWS VPC Peering service as well as the other practical issues described in **section 1.2**, the architecture design of GoToCloud permits the creation of multiple mirror EFSs. These additional EFSs are again placed in their own VPCs (e.g., "MirrorA2 of Shared EFS" in "MirrorA2 VPC" as the second set) within a single AWS Region (e.g., A Region). In this way, more than 125 users can be supported (e.g., "User126 AWS account" having "User126A VPC"). Furthermore, a pair of mirror EFS and VPC (e.g., "User001B VPC" in "User001 AWS Account") can be created in the

user's preferred AWS Region (e.g., "B Region") and connected to the GoToCloud management group's VPC (e.g., "MirrorB1 of Shared EFS" in "MirrorB1 VPC") in the same AWS Region by a VPC Peering connection, to implement the multi-region support.

4. Supplementary References

1. Zivanov, J., Nakane, T. & Scheres, S. H. W. Estimation of high-order aberrations and anisotropic magnification from cryo-EM data sets in RELION-3.1. *IUCrJ* **7**, 253–267 (2020).
2. Kimanius, D., Dong, L., Sharov, G., Nakane, T. & Scheres, S. H. W. New tools for automated cryo-EM single-particle analysis in RELION-4.0. *Biochem. J.* **478**, 4169–4185 (2021).
3. Pettersen, E. F. *et al.* UCSF Chimera - A visualization system for exploratory research and analysis. *J. Comput. Chem.* **25**, 1605–1612 (2004).