Analysis of the duplicate file problem at LDCS

Department of Particle Physics

Piotr Yartsev

Summer & Autumn, 2022

Project duration:
Full time
2 months

Supervisor:
Ruth Pöttgen

Abstract

In this work, we aim to research the problem of duplicate files being generated during particle physics simulation using the Lightweight Distributed Computational System (LDCS) [3]. This project is a continuation and is based on the findings made in the bachelor thesis project "Improvement of the Rucio implementation for the LDCS platform and search for dark data" [4]. In the course of this project, we were able to disprove several preexisting theories for the source of the duplicate files, generate new data studying the duplicate file phenomenon, and based on this data formulated a new theory in an attempt to explain the mechanism responsible for these duplicate files.

Contents

1 Introduction

The Light Dark Matter eXperiment (LDMX) [1] is a proposed accelerator-based experiment to search for light-dark matter particles. At this stage, LDMX is undergoing feasibility and design studies which require massive computer simulations and complex digital infrastructure to handle, analyze and store all the data generated. To do that the Lightweight Distributed Computing System (LDCS) [3] was created and it is using a range of different hardware and software.

As the name suggests, LDCS is a distributed system, meaning it uses a multitude of computational centers, file storage locations, and other digital infrastructures spread around the world. That means that digital operations, such as performing a particle physics simulation, have to be sent to different computational sites and the outputs have to then be moved to the right storage location. A problem was discovered with the simulation workflow as at times a single submitted simulation job would output multiple files, something that should not be possible. To find these duplicate files, as well as other problematic files, was the aim of my 2022 bachelor thesis.

In the Spring 2022 thesis paper "Improvement of the Rucio implementation for the LDCS platform and search for dark data" [4], which for the rest of this paper will be referred to as **the thesis paper** or **the thesis project**, the dark data problem was addressed by the creation of the Dark Data Search (DDS) toolkit, which was designed to discover and categorize the different dark data files in storage for the LDMX experiment. After a surface-level study, several questions remain about the source of these duplicate files, which is something this project aims at answering by analyzing the output generated by the DDS toolkit as well as other available data.

2 Background

In the thesis paper, several discoveries about the dark data situation at LDCS were made, including discoveries about the problem with duplicate files. By duplicate files, we are referring to files that share the Filename Before Timestamp (FBT), which is normally supposed to be unique for every file in a simulation job [2].

$$\underbrace{\underbrace{\text{mc\_\_v9-8GeV-1e-target\_\_photonuclear}}_{\text{Unique for every simulation run}}\_\_\overbrace{14569}^{\substack{\text{Unique number for each}\\ \text{file in simulation run}}}}_{\text{Filename Before Timestamp (FBT)}}\_\underbrace{t1589280908}_{\text{timestamp}}.\text{root} \quad (1)$$

Using the DDS toolkit we were able to find and classify dark data files on the storage used by LDCS. In table [1] we can see that the duplicate files are responsible for the vast majority of the files that were present in storage but were not registered to the cataloging system Rucio. The exception to this is the storage at Lund without GridFTP access, which in the rest of this paper will be referred to as just the storage in Lund, which had a roughly 50/50 split between duplicate and non-duplicate files in that category.

Table 1: State of dark data problem on LDCS storages
Table showing the state of the dark data situation at three of the storages used by LDCS. The values here are only for files that exist in storage but are not registered in the file cataloging service Rucio. For this project what is of interest is third row which shows the number of duplicate files found at the different storage's using the DDS toolkit.

	LUND	LUND_GRIDFTP	SLAC_GRIDFTP scope mc20
Missing from Rucio	1052	202454	1744
Not a duplicate	439	661	5
Duplicate	589	201673	1699
Problem simulation runs	2	3	0

Due to "files missing from Rucio" being by far the largest fraction of dark data files [1], finding the source for these duplicate files could save a substantial amount of storage space. As the creation of duplicate files is most likely related to the submission and/or simulation software used by LDCS, finding the source of the duplicate files problem could unveil a more fundamental problem with the current simulation workflow.

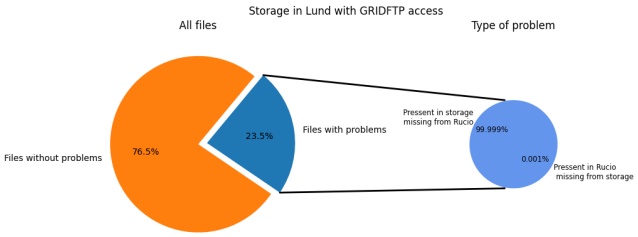


Figure 1: Distribution of dark data files at storage in Lund
The left pie plot shows all the files registered and located at storage at Lund with remote GridFTP access. The blue section of the left pie represents all files that were flagged as dark data and the distribution of the type of dark data can be seen in the right pie plot.

To study this further, an attempt was made to generate these duplicate files by running a small simulation campaign using the LDCS infrastructure. This attempt was successful and we discovered duplicate files when one of the batches of simulations completed 93 simulations jobs, but the output folder contained 111 files. These duplicate files were compared to each other and it was found that they were all valid particle physics simulation output files and comparing two such duplicates files showed they contained very similar values with some small differences. The plot showing the timestamp for the regular and duplicate files can be seen in figure [2]. In the plot, we can see that the files seem to follow two trends based on the timestamp and the frequency of arrival for the output files.

Based on this result it was theorized that the source for these duplicate files had to do with the same simulation/reconstruction job being sent at two different computational centers which then in turn both attempt to create and register an output file in the Rucio catalog, with only one succeeding.

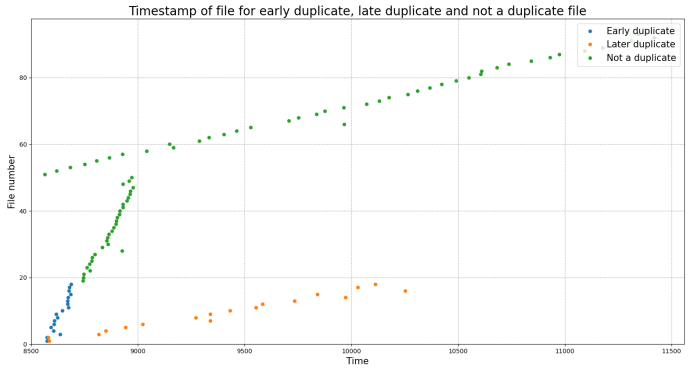


Figure 2: Timestamp vs file number for duplicates and regular files Plot showing the relations between the timestamp, on the x-axis, for the early duplicate (blue dots), later duplicate (orange dots), and the timestamp for the files with no duplicates (green dots) against the file number on the y-axis.

After performing a surface-level analysis of the duplicate files created during the thesis project simulation campaign it was agreed that further research would be needed. Therefore it was decided to test the proposed theory and attempt to find the source for these duplicate files, which is the aim of this project.

3 Encountered problems

3.1 Too many duplicates and superdirectories

To test the tools we developed as part of this project, which is further discussed in the section "Method for generating the database" 10, we ran it for a manually selected collection of directories from the storage at Lund and Lund with GridFTP access, which for the rest of this paper will be referred to as Lund GridFTP. The choice to look at files at both storages at the same time was made because the prominent theory at that time was that duplicates were created due to the proximity of the storage location to the computational center, which is something this paper was not able to prove or disprove. Both the storages are located at Lund University, so there should not exist a difference between them in the case that this theory was correct. The first run can be seen in figure [16] and we can see that the files are clustered together, so the clusters were numbered 1-7 for easier analysis.

During the analysis of this limited run, the result from cluster 6 caught our attention. In this cluster, see figure [18], we can see the files showing many different patterns at the same time, such as overlaying trends of files. While this does not prove anything, and it was later shown that this behavior was due to it being test data and not due to the superdirectory problem, we decided to look into the files in this cluster. After studying the files in this cluster we found that while those files were all located in the same directory, they belonged to many different datasets.

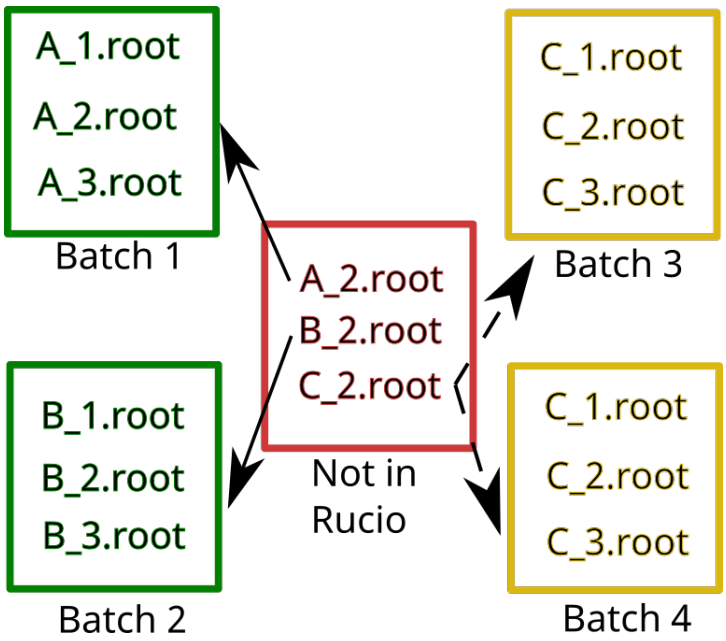


Figure 3

The problem is visualized in the figure 3 above where we have sets of files belonging to dataset **Batch 1**, **Batch 2**, **Batch 3**, and **Batch 4**. We also have several files that are not registered in Rucio, meaning we do not know what dataset they belong to. While we can't check Rucio for what dataset those files belong to, we can check if the FBT of those dark data files only matches the FBT of a known batch, as can be seen in figure 3 where **A__2.root** and **B__2.root** belong to **Batch 1** and **Batch 2** respectively. But this still leaves the problem with dark data files that share FBT with two datasets, as shown by the file **C__2.root** which shares FBT with both **Batch 3** and **Batch 4**, in which case it is much harder to know what dataset they belong to. While there could exist a method of matching those dark data files to datasets by looking at the timestamp and other available file data, due to the large number of files the computational time needed to separate them is beyond what we can dedicate to this problem.

This leads to the conclusion that the duplicate file problem could be overestimated by the DDS toolkit, as discussed in [6.5]. While this can be corrected in the next version of the DDS toolkit the superdirectory problem presents a big challenge for the current project, which is what to do with the "not in Rucio" files that we can not determine the dataset for. It was decided to remove these files from the analysis, which predominately impacts the storage at Lund GridFTP, meaning that going forward in this paper the results for the number of duplicate files is the lower limit.

3.2 Test and validation data

In this project, we encountered many datasets that contained a much higher number of duplicates as well as much longer duplicate chains, which is the collection of duplicates sharing a single FBT, than what was expected. An example of such a dataset can be seen in figure [4]. A common factor between them was that they all contained the word "test" in the name of the dataset.

After looking into those datasets further and with the help of Lene Kristian we verified that these datasets were generated during a test of the system and are not representative of simulations that we would run for the LDMX experiment. It was also noted by Lene Kristian that those datasets could belong to the scope **validation**, and after looking into it further we found that a number of the datasets also belonged to scopes containing the word "test", such as **test-v2.1**.

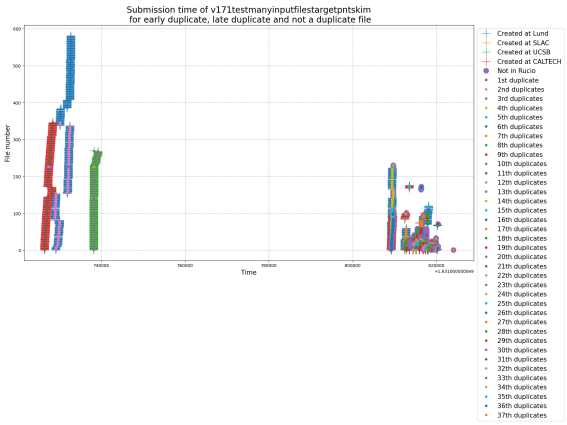


Figure 4: Long chains for datasets in scope "test-v2.1"
A plot showing the time of the file's creation against the file number with different markers to denote what computational center they were created at and what, if any, duplicate number they have. This dataset shown is an example of a dataset with many long duplicate chains, in this case, up to 37 duplicates for a single FBT, that contain "test" in its name and belong to the scope **validation**.

It was decided not to include these datasets in our analysis because these files are not representative of the actual simulation of the LDMX or even the LDCS setup, as it is often used to test some features that might not be properly implemented. To remove such datasets we wrote a program that looks for datasets containing files belonging to the scope **validation** or a scope containing the word "test", which we collectively call **bad scopes**. We then ran the program cleaning the database in three different modes:

1: Removing any datasets consisting exclusively of files in the problem scopes

2: Removing all files belonging to the problem scopes

3: Leave all files in the problem scopes untouched

From the result in table 6, table 5 and table 7 in combination with manually studying what files and datasets were removed by modes 1 and 2, it was decided that using mode 2 resulted in data that was the most representative of what would be generated by an actual simulation campaign.

3.2.1 Shared dataset and missing data

After running the cleaning tool some discrepancies were found between mode 1 and mode 2. It would be expected that there would not be any difference between the two modes, as the only situation where data would remain untouched by mode 1 but get removed by mode 2 would be if we had datasets in our database with more than one scope, which was something believed not to be possible and fundamentally goes against how Rucio operates. A theory that could explain this result was that two datasets that share the same name, although registered to different scopes and therefore not the same dataset, were incorrectly grouped in the database. While we did include safeguards against that, they might have failed to detect such datasets.

To test this theory we decided to study the datasets removed by mode 2 that were untouched by mode 1 and found several examples such as can be seen in figure 5.

	file	BatchID	Scope	duplicate	file_number
	Search column...	v2.3.0-batch1	Search column...	Search column...	899
1	mc_v12-4GeV-1e-ecal_photonuclear_run899_t1614651522.root	v2.3.0-batch1	validation	2	899
2	mc_v12-4GeV-1e-ecal_photonuclear_run899_t1608081924.root	v2.3.0-batch1	mc20	1	899

Figure 5: Many datasets in one table
A image showing two files in our database that are duplicates of each other, sharing the same FBT, both belonging to the same dataset, **v2.3.0-batch1**, but showing different scopes, **mc20** and **validation**

We looked in the Rucio catalog for these two combinations of scope and dataset, **mc20:v2.3.0-batch1** and **validation:v2.3.0-batch1**, and as shown in figure 6a and 6b both of these combinations do exist. Next, we checked if the files were stored in the same location, in which case this could be a superdirectory.

DATASET: mc20:v2.3.0-batch1

RSE	FOUND	TOTAL
CALTECH	1569	4822
SLAC	100	4822
LUND	1501	4822
UCSB	1652	4822

(a) An image showing the distribution of files for the combination of scope and dataset **mc20:v2.3.0-batch1**.

DATASET: validation:v2.3.0-batch1

RSE	FOUND	TOTAL
LUND	2112	4998
LUND_GRIDFTP	4998	4998
UCSB	1823	4998

(b) An image showing the distribution of files for the combination of scope and dataset **validation:v2.3.0-batch1**.

To verify the location of these datasets we extracted the storage directory from Rucio for the two files shown in figure 5 and we saw that they were stored in different directories:

mc20: /ldmx/mc-data/**mc20**/v12/4.0GeV/v2.3.0-batch1/mc_v12-4GeV-1e-ecal_photonuclear_run899_t1608081924.root

validation: /ldmx/mc-data/**validation**/v12/4.0GeV/v2.3.0-batch1/mc_v12-4GeV-1e-ecal_photonuclear_run899_t1614651522.root

While this problem could be corrected retrospectively, another problem was discovered with the database where some datasets were missing that should not have been, and it was unclear at which point in the process they were incorrectly removed. This forced us to remake the database from the output data from the DDS toolkit, which takes several days in just computational time, and had us redo all analysis using the new corrected data.

Any discrepancies left between mode 1 and 2 left in table 6, table 5, and table 7 are due to an error in the software which let some problem scopes trough using mode 1. After manually analyzing the database cleaned using the different modes it was verified that the data cleaned using modes 2 and 3 is correct. Because we will for all future analyses only use the data cleaned by mode 2 it was decided not to use project time to correct the error with mode 1, but it means the values in the middle column can not be trusted for all three tables.

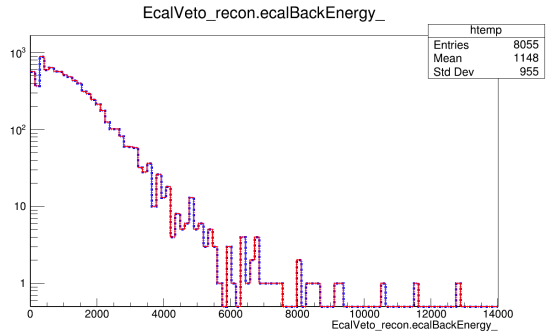


Figure 7: Comparing the content of two duplicates
Overlay of two plots for the duplicate with file number 1 generated during the simulation campaign as part of the thesis project. The earlier arriving duplicate file is marked in a blue dotted line and the later arriving duplicate is marked in a red solid line.

4 Theories tested and disproved

4.1 Examining failed jobs as the source of a duplicate file

A prominent theory discussed during the thesis project was the possibility of the source of the duplicate files having to do with "fake" failed simulations. The general idea was as follows:

I: A job would falsely indicate to ACT that it had failed to begin simulation or that it had failed during simulation, while in reality, it was able to complete simulations correctly.

II: The ACT would mark that simulation as failed and put that job back in the job queue for an additional simulation attempt.

III: While the first job completed successfully it generated a output file, due to ACT believing the job failed it would not be able to generate correct metadata and/or registering the file in Rucio

IV: The resubmitted job would either falsely report failing again, generating a second duplicate file, or be correctly registered as a successful simulation and be correctly registered in Rucio.

Theories of this type had several advantages. It could explain why ACT did not report the number of jobs running correctly, as it would not be unreasonable to think that once the job was falsely registered as failed ACT would remove it from the list of running jobs, even if it was running.

It could also explain the fact that two duplicates would contain very similar output data, as can be seen in figure [7] where we compare the same plot between two duplicate files, as it would have been the same job with the same seed resubmitted and the small differences could be explained by the small amount of randomness that we expect from the simulations.

The last advantage is that this source of duplicate files would have been easy to verify, as we could count the number of jobs that failed and compare it to the number of duplicate files. If the number of failed jobs was equal to or larger than the number of duplicates we considered it evidence that this was a possible source that had to be researched further.
What was found was that the number of failed jobs did not match or exceed the number of duplicate files, and in some cases, no failed simulation jobs were reported for a dataset containing duplicate files. Based on these findings we rule out this theory as the possible source for the duplicate files problem.

4.2 Reconstructions

During the simulation campaign, as part of the thesis project, it was noted that we saw more duplicate files outputted from reconstruction jobs compared to simulation jobs. A theory that could explain this phenomenon would be related to failed jobs being the source of the duplicate files, as reconstruction jobs could be failing more frequently than simulation jobs, and hence they would be expected to contain more duplicate files. The theory of failed jobs being the source for the duplicate files problem was disproved, and while there existed no concrete alternative theory on what mechanic could cause these discrepancies, it was still deemed worthy of looking into.

Table 2: Duplicate files at simulations and reconstructions
Table showing the state of the duplicate file situation for simulation datasets and reconstruction datasets. "Duplicate percentage of simulation" and "Duplicate percentage of reconstructed" show the percentage of files for a simulation or reconstruction job that were duplicate. "Percentage of simulations with duplicates" and "Percentage of reconstructions with duplicates" show the percentage of simulation datasets and reconstruction datasets that contain at least a single duplicate file.

	LUND	LUND_GRIDFTP	SLAC_GRIDFTP scope mc20
Duplicate percentage of simulation	0.037%	38.7%	2.7%
Duplicate percentage of reconstructed	0.035%	15.2%	2.71%
Percentage of simulations with duplicates	22.9%	94.7%	100.0%
Percentage of reconstructions with duplicates	37.0%	98.0%	100.0%

From the results in table 2, as well as from manual analysis of the duplicate files at simulation datasets and reconstruction datasets, we concluded that there does not seem to be any significant difference between simulation datasets and reconstruction datasets. The only contrary example is the storage at Lund GridFTP where the duplicate file problem seems to be more prevalent in simulation data than in reconstruction data.

4.3 Metadata comparison between regular and duplicate files

One avenue we pursued was to look at the metadata in Rucio of duplicate files and compare it to the metadata of regular files. The idea was, that perhaps there was some specific setting or configuration that was applied to the duplicate files causing the, or increasing the probability for, duplicate file problem. After comparing the metadata of all duplicate files, examples of which you can see in figure 19, figure 20 and figure 21, no metadata attribute was found to be unexpected and the occurrence did not significantly different from that of regular files.

5 Results

5.1 State of duplicate file problem

The purpose of this analysis was to get a general understanding of the state of the duplicate file problem at the storage used by LDCS.

Table 3: Summery of the duplicate files situation at all three locations
Table showing the state of the duplicate file situation at the storage at Lund with and without GRDFTP access and for the storage at SLAC, but only for files registered to scope mc20. The "The number of files" row shows the total number of files analyzed in storage, the "number of duplicates" shows the number of duplicate files we found and the "percentage of duplicates" shows how large part of the total number of files analyzed were duplicate files. The row "Number of duplicates after removing the first" shows the duplicate file situation if we assume that one file per duplicate file chain is the original "good" file, hence we remove one duplicate from each chain. The "Percent of duplicate after removing the first" shows how large part these assumed "bad" duplicates take up and the row "Longest chain" shows what is the longest chain of duplicates sharing the same FBT among all the files at that storage location.

	LUND	LUND_GRIDFTP	SLAC_GRIDFTP scope mc20
Number of files	406461	875726	45618
Number of duplicates	151	338930	1234
Percent of duplicates	0.0%	38.7%	2.7%
Number of duplicate after removing the first	76	236977	626
Percent of duplicate after removing the first	0.0%	27.1%	1.40%
Longest chain	3	19	3

Table 4: Duplicate file timeline
Table showing the time frame, start to finish, of the regular files and the duplicate files. The time frame is calculated using the timestamp of the files, meaning it is when the output file is generated, not when the job was submitted. We can see that for the storage at SLAC and the storage at Lund GridFTP the time frame for all files and the time frame for duplicate files almost completely overlap, while for the storage at Lund access we see that the duplicate files were only generated during a subsection of the entire time frame analyzed.

	Regular files time frame	Duplicate files time frame
Lund	2020-05-10 02:16:53 - 2021-02-05 02:11:43	2020-09-26 03:56:19 - 2020-12-29 16:22:12
Lund GridFTP	2021-03-15 18:09:40 - 2022-05-06 04:02:40	2021-03-15 18:09:52 - 2022-05-06 04:02:40
SLAC GridFTP scope mc20	2020-12-26 11:04:58 - 2020-12-29 22:30:26	2020-12-26 11:04:58 - 2020-12-29 22:15:17

5.2 Cleaning mode result

The purpose of this analysis was to confirm that the cleaning using the different modes worked correctly, which was shown to not be the case for mode 2, as well as to get a better understanding of how the duplicate problem affects actual simulation data and how it affects test data.

Figure 8: For the three tables below, the column corresponds to the methods described in the list [3.2], with the first column showing the values as described by mode 3, the second column as described by mode 1 (the values are incorrect and should not be used for analysis) and the third column as described by mode 2. The first row shows the number of the first duplicate in duplicate chains, which would also signify the number of duplicate chains. The second row shows the total number of duplicate files found. The third row shows the percentage of all duplicates that are the first duplicate. The fourth row shows the percentage of total duplicate files remaining after each cleaning step compared with the number cleaned using mode 3.

Table 5: Data from Lund cleaned by mode 1, 2 and 3
The duplicate file situation at the storage at Lund, with the structure as described in 8

Lund			
	All datasets	Removed datasets consisting only of test/validation	Removed all occurrence's of test/validation
Total number of first duplicate	2236	524	75
Total number of all duplicates	4992	1298	151
Percentage of first duplicate	44.8%	40.4%	49.7%
Compared to the total number in all datasets	100%	26.0%	3.02%

Table 6: Data from Lund with GridFTP cleaned by mode 1, 2 and 3
The duplicate file situation at the storage at Lund GridFTP, with the structure as described in 8

Lund with GridFTP access			
	All datasets	Removed datasets consisting only of test/validation	Removed all occurrence's of test/validation
Total number of first duplicate	103022	101955	101953
Total number of all duplicates	343681	338938	338930
Percentage of first duplicate	30.0%	30.1%	30.1%
Compared to the total number in all datasets	100%	98.6%	98.6%

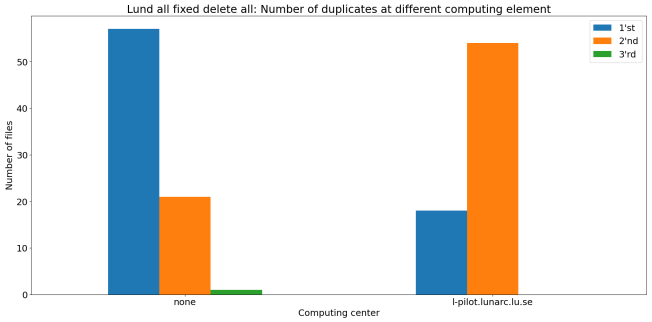
Table 7: Data from SLAC cleaned by mode 1, 2 and 3
The duplicate file situation at the storage at SLAC for only the files belonging to scope mc20, with the structure as described in 8

SLAC only scope mc20			
	All datasets	Removed datasets consisting only of test/validation	Removed all occurrence's of test/validation
Total number of first duplicate	608	608	608
Total number of all duplicates	1234	1234	1234
Percentage of first duplicate	49.3%	49.3%	49.3%
Compared to the total number in all datasets	100%	100%	100%

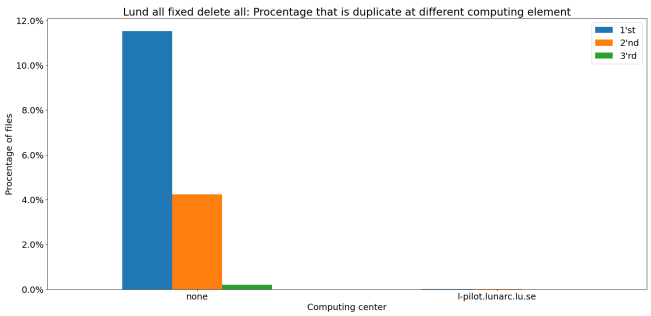
5.3 Duplicate distribution

The purpose of this analysis was to get an understanding of the distribution of duplicate files among the computational centers to study the theory that duplicate files originate from the same job being sent to different computational centers. We also analyze the distribution of duplicate files within a dataset to look for patterns that could be exploited in the search for the source of the duplicate file problem.

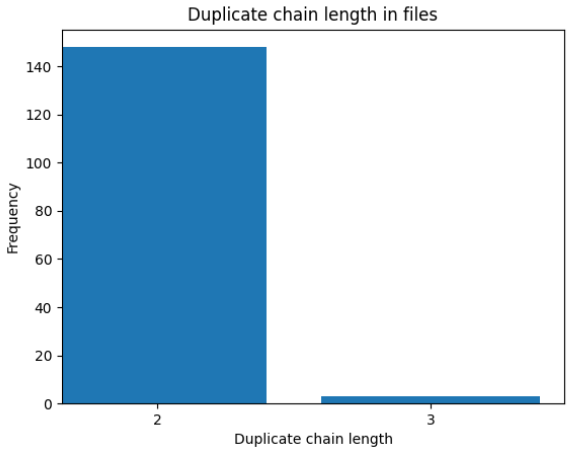
5.3.1 Lund



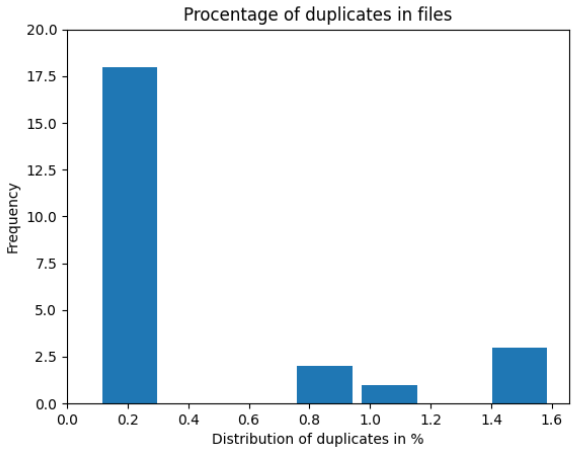
(a) Lund: Distribution of duplicates among computational centers
Number of duplicates generated at different computational centers, or assigned "None" if no record of them exists in Rucio. The duplicate number as denoted by the different colors is signifying the relative time of creation compared to the other duplicates with the first duplicate created receiving number 1, the second number 2, etc.



(b) Lund: Percentage duplicates for computational centers
Percentage of all files generated by a computational center, or assigned "None" if no record of them exist in Rucio, that are duplicates. The duplicate number as denoted by the different colors is signifying the relative time of creation compared to the other duplicates with the first duplicate created receiving number 1, the second number 2, etc.

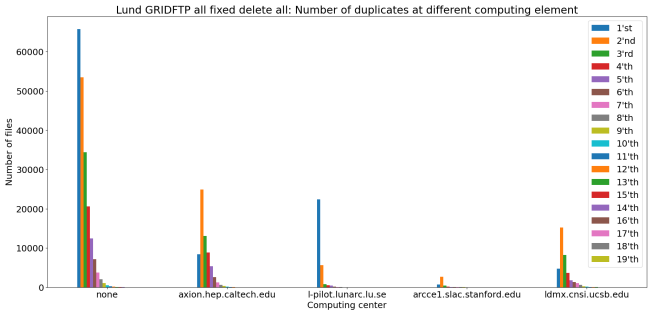


(c) Lund: Distribution of duplicate chains length
Bar chart showing the occurrence of different lengths of duplicate chains found at the storage in Lund.



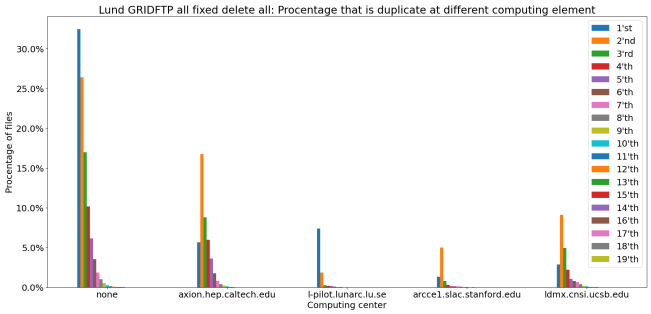
(d) Lund: Percentage of a dataset that are duplicate files
Bar chart showing the occurrence of the percentage of a dataset that is duplicate files for the storage in Lund.

5.3.2 Lund with GridFTP



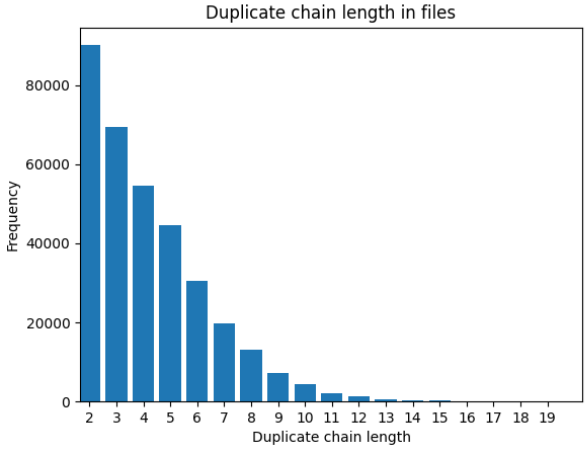
(a) Lund GridFTP: Distribution of duplicates among computational centers

Number of duplicates generated at different computational centers, or assigned "None" if no record of them exists in Rucio. The duplicate number as denoted by the different colors is signifying the relative time of creation compared to the other duplicates with the first duplicate created receiving number 1, the second number 2, etc.



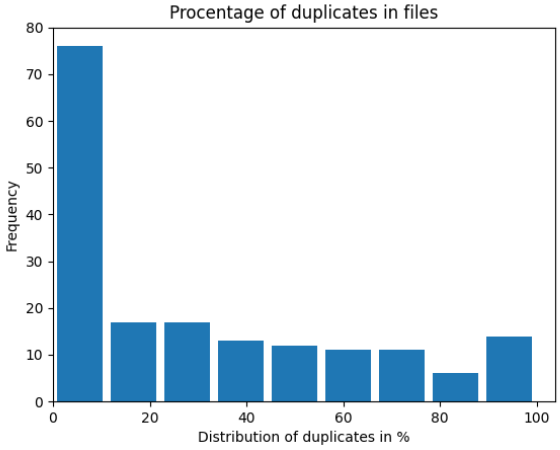
(b) Lund GridFTP: Percentage duplicates for computational centers

Percentage of files generated by a computational center, or assigned "None" if no record of them exist in Rucio, that are duplicate. The duplicate number as denoted by the different colors is signifying the relative time of creation compared to the other duplicates with the first duplicate created receiving number 1, the second number 2, etc.



(c) Lund GridFTP: Distribution of duplicate chains length

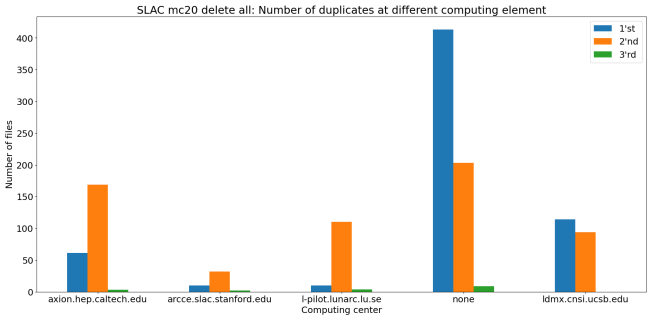
Bar chart showing the occurrence of different lengths of duplicate chains found at the storage at Lund GridFTP.



(d) Lund GridFTP: Percentage of a dataset that are duplicate files

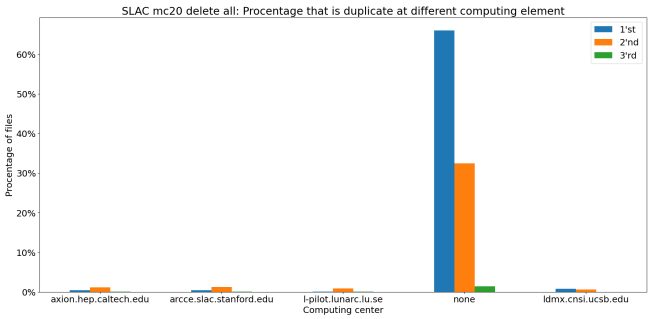
Bar chart showing the occurrence of the percentage of a dataset that is duplicate files for the storage at Lund GridFTP.

5.3.3 SLAC scope mc20



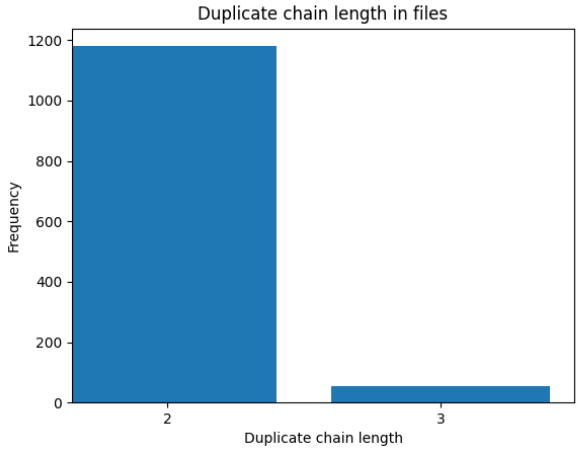
(a) SLAC: Distribution of duplicates among computational centers

Number of duplicates generated at different computational centers, or assigned "None" if no record of them exists in Rucio. The duplicate number as denoted by the different colors is signifying the relative time of creation compared to the other duplicates with the first duplicate created receiving number 1, the second number 2, etc.



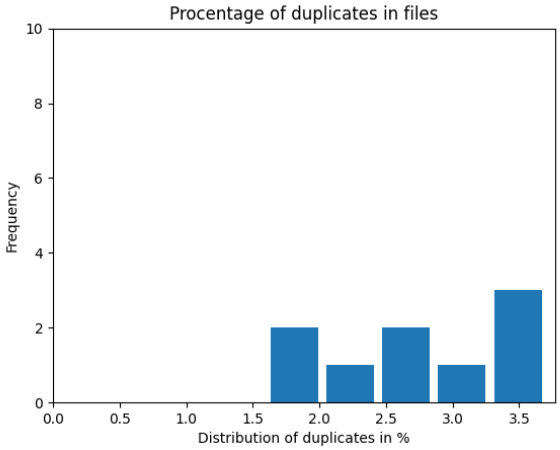
(b) SLAC: Percentage duplicates for computational centers

Percentage of files generated by a computational center, or assigned "None" if no record of them exist in Rucio, that are duplicate. The duplicate number as denoted by the different colors is signifying the relative time of creation compared to the other duplicates with the first duplicate created receiving number 1, the second number 2, etc.



(c) SLAC: Distribution of duplicate chains length

Bar chart showing the occurrence of different lengths of duplicate chains found at the storage at SLAC.



(d) SLAC: Percentage of a dataset that are duplicate files

Bar chart showing the occurrence of the percentage of a dataset that is duplicate files for storage at SLAC.

5.4 Results: Duplicate chains longer than 2

We see at the storage at Lund without GridFTP access and at the storage at SLAC all duplicate chains, length 2 or longer, consisting of one file registered in Rucio and the rest missing from the Rucio catalog, as can be seen in the example figure 12. For the storage, at Lund GridFTP, we see in the example figure 13 that more than one duplicate per chain can be registered to Rucio.

file	ComputingEle...	file_number	duplicate
Search column...	Search column...	1283694	Search column...
mc_v12-4GeV-1e-ecal_photonuclear_run1283694_t1601306683.root	NULL	1283694	3
mc_v12-4GeV-1e-ecal_photonuclear_run1283694_t1601306529.root	NULL	1283694	2
mc_v12-4GeV-1e-ecal_photonuclear_run1283694_t1601306461.root	l-pilot.lunarc.lu.se	1283694	1

Figure 12: Example duplicate chain for storage at Lund
An image showing the files belonging to the duplicate chain of length 3 for file number 1283694. In the column "ComputingElemnt" Null indicates that the information about where the file was generated is missing, in this case, due to the file being missing from Rucio.

file	ComputingEle...	file_number	duplicate
Search column...	Search column...	856	Search column...
mc_v9-8GeV-1e-target_photonuclear_run856_t1631833999.root	l-pilot.lunarc.lu.se	856	1
mc_v9-8GeV-1e-target_photonuclear_run856_t1631837919.root	NULL	856	2
mc_v9-8GeV-1e-target_photonuclear_run856_t1631837926.root	axion.hep.caltech.edu	856	3
mc_v9-8GeV-1e-target_photonuclear_run856_t1631840440.root	NULL	856	4
mc_v9-8GeV-1e-target_photonuclear_run856_t1631840519.root	NULL	856	5
mc_v9-8GeV-1e-target_photonuclear_run856_t1631840604.root	ldmx.cnsi.ucsb.edu	856	6

Figure 13: Example duplicate chain for storage at Lund GridFTP
An image showing the files belonging to the duplicate chain of length 6 for file number 856. In the column "ComputingElemnt" any value but Null indicates that we have information about where the file was generated, meaning the file exists in the Rucio catalog.

The purpose of this analysis is to better understand how duplicate files in a single chain are related to each other by comparing metadata between duplicates registered in Rucio.

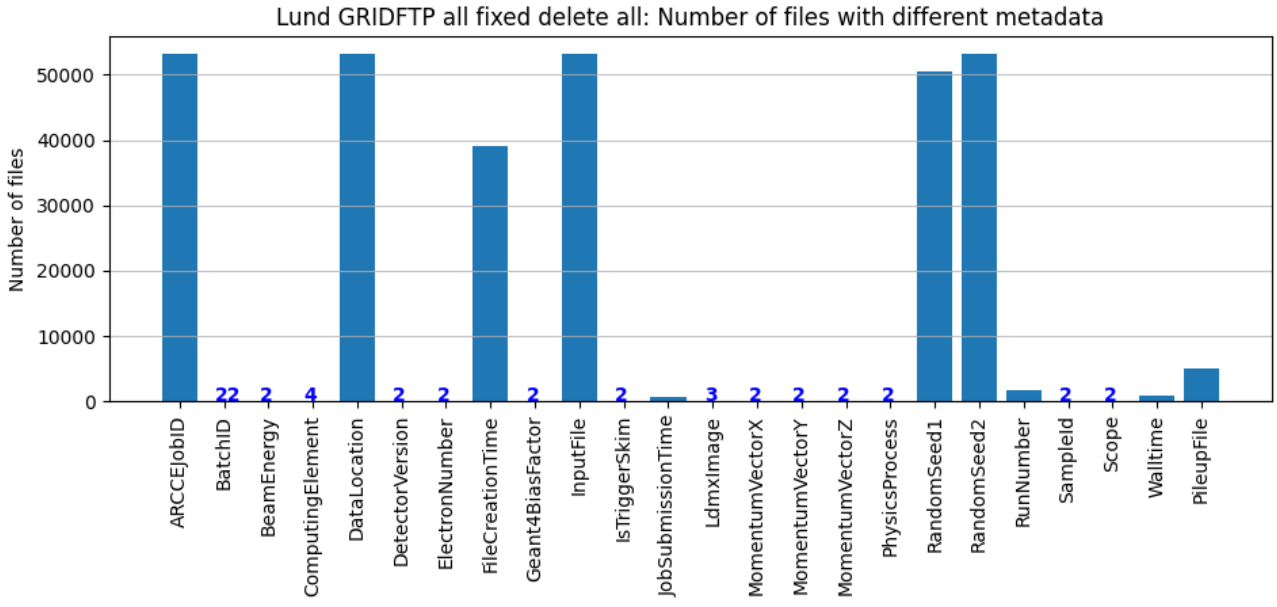


Figure 14: Metadata differences between duplicates in the same chain
A plot showing the occurrence of different metadata attributes that differ between two or more duplicate files from the same duplicate chain. All values less than 500 are displayed with a number in blue text.

6 Discussion

6.1 Percentage of files that are duplicate

From table 4 we can see a fundamental difference between the storage at Lund and SLAC compared to the storage at Lund GridFTP. While the difference between the percentage of files that are duplicates at Lund and SLAC is an order of magnitude, the duplicates occupy a relatively small portion of the total storage and are at a level I would consider acceptable and would not be worth putting in significant time mitigating. Where we see a dire picture is the storage at Lund GridFTP with between 27.1% and 38.7% of the files being duplicates, depending on if you believe the theory that one file per chain is proper simulation data or not.

Due to the result being so extreme for the storage at Lund GridFTP a large effort was dedicated to verifying this result. One such verification method was to rerun the search for duplicate files at that storage using many different methods and algorithms to see if an error in the main algorithm resulted in an overestimation of the problem. All of these algorithms resulted in the same outcome. We also verified that all the duplicate files in all the datasets exist in the datasets directory, and no missing files were found among the 338930 duplicate files. Based on the verification not showing any discrepancies, I have to assume that the result is correct.

6.2 Cleaning results

From the tables, 6 5 7, showing the result of cleaning the data using the different modes we see that "test" data is not the main source of the duplicate file problem for all the storages but Lund. This is an important result as it was discussed that it would not be unexpected to see duplicates in "test" data because those files can be generated using very experimental LDCS setups and would not have to follow proper naming schemes for the files, which is what we use to find duplicates. Based on our findings we can conclude that the duplicate file problem, which is the most prevalent for the storage at Lund GridFTP, is present in real simulation data meaning there exist a problem with LDCS.

As mentioned earlier, the exception to this is the storage at Lund which loses 96.8% of the duplicate files after removing all files in scopes containing the word "test" and "validation". This seems to indicate that while the idea proposed for the duplicate file source being "test" data could be true, it is not the main source.

6.3 Duplicate distribution

Here we again see a difference between the duplicate file situation at the storage at Lund and SLAC compared to the storage at Lund GridFTP. As we can see in figure 9a and figure 11a that the number of first duplicates missing from Rucio (None) at the storage at Lund and SLAC matches the sum of all the second duplicates at all the other locations if you account for the duplicate with chains longer than 2.

This in combination with the result from figure 11c and 9c and the observations we made in section 5.4 would collaborate a theory we discussed during the thesis project for how the mechanism that generates these duplicate files could work:

- LDCS would correctly generate one output file for one simulation job and correctly register it in Rucio
- It would also, due to some unknown error, generate one or more extra files and fail to register them in Rucio

This theory had a big advantage in that we could delete all the files missing from Rucio and keep one good output file for every simulation job.

What disproves this theory, or at least shows that the theory does not apply to all the storages, is the situation at the storage Lund GridFTP. At this storage, the number of first duplicates that are not registered in Rucio (None) does not match the sum of all second duplicates at the other computational centers, even after we account for the length of all the duplicate chains. This can be explained by the existence of multiple duplicate files registered in Rucio in a single duplicate chain, as can be seen in figure 13.

This poses a problem for the theory as it assumes that when we submit a computational job, simulation or reconstruction, we correctly output one "good" file, registered in Rucio, and some number of "bad" files, which are not registered in Rucio. For the storage at Lund GridFTP, we have multiple "good" files in one duplicate chain, meaning we would not know which, if any, is the original correct output.

Another thing of interest is that the distribution of duplicate chain lengths for the storage Lund GridFTP seems to follow an exponential decay curve, as can be seen in figure 10c. If this is the case, it could mean that the source for the duplicate files is probabilistic, meaning for each simulation job we have a certain chance to run it again, generating a duplicate file, and those additional simulation jobs themself have the same probability to create an additional simulation job, et cetera.

I do not have a good enough understanding of the LDCS setup to come up with what could cause such a probabilistic source. It is possible that the storage at Lund 9c and storage at SLAC also follow a similar decay curve but with only two data points (chains of length 2 and chains of length 3) we can not say anything for sure.

6.4 Comparing metadata for duplicates in the same chain

Because the storage at Lund GridFTP has multiple files in a single duplicate chain that are registered in Rucio we can compare metadata between them, which we do in figure 14. Some of the metadata attributes being different between the duplicates are not surprising, such as the **ARCCEjobID**, as all simulations jobs should have different ARC job Id, as well as **DataLocation** and **FileCreationTime**, as all the duplicates have had different timestamp which in turn would change the filename, which is part of the **DataLocation** attribute.

What was unexpected was the difference in the attribute **InputFile**, **RandomSeed1**, and **RandomSeed2**. This indicates that the duplicates are not reruns of the same simulation job, but are completely different simulation jobs entirely. This goes directly against what was the prominent theory before this project which was that duplicate files were generated by rerunning the same simulation using the same simulation parameters, resulting in the file content of the duplicates being similar to each other, as can be seen in figure 7 and discussed in the thesis paper [4].

Another important result is that the attribute **JobSubmissionTime** rarely differs between duplicates from the same chain. This means the duplicates, which we showed are not reruns of the same job but output from completely different simulation jobs, were submitted at the same time. Based on this fact, as well as everything else we have discussed in this paper, I theorize that the source of these duplicate files is as follows:

- Several different unrelated computational jobs are submitted at the same time.
- The system at some point in the simulation workflow incorrectly assumes that all of these jobs are the same job, and therefore uses the same naming scheme for all of them giving them the same FBT.
- This happens before the timestamp is added to the filename, as it differs between duplicates, which somewhat limits where in the workflow we have to search for the error.

As we lack the metadata from the files not registered in Rucio, I can not say if this theory applies to them or not. This theory could also explain the difference in the percentage of storage that is duplicate files, as substantially more computational jobs are submitted to then be put at the storage at Lund GridFTP, meaning the chance of two **JobSubmissionTime** matching would increase for them.

What this theory does not explain is why the distribution of duplicate chains would follow an exponential decay curve and, more importantly, does not explain why some duplicates are registered in Rucio while others are not. I believe some additional research would be needed by LDCS to test this theory and to get a deeper understanding of the mechanics that cause this problem.

6.5 Problems found with DDS-toolkit and some solutions

During this project, several problems were found regarding the DDS toolkit that was developed during the thesis project.

6.5.1 Super-directories

A large number of datasets with duplicate files shared a single directory, as discussed in this section [3.1]. This is something that the DDS toolkit did not account for when we analyzed the dark data situation at LDCS. This means that the DDS toolkit could find files sharing an FBT, and mark them as duplicates, even though they all belong to different datasets and should not be counted. Therefore we can not trust the value for the number of duplicates found or the number of duplicates for a single FBT. The extent of this error has not been estimated.
During this project, a solution was developed for us to find the duplicate files which work very well and is substantially faster than the old solution. This solution has yet to be integrated into the DDS toolkit, but doing so should not take much effort.

6.5.2 Wrongly assigned scopes

In the DDS toolkit the files missing from Rucio, which then would not have metadata associated with them, were assigned a scope and a dataset based on the directory they belonged to, but there were two problems with that solution:

I: This fails to take into account the problem with super-directories.

II: A error in the way the DDS-toolkit stored that data resulted in all files missing from Rucio taking the values for the dataset and scope from the last processed file that exists in both the Rucio catalog and the storage.

III: The format for the data in the output .txt for the files missing from Rucio was not consistent with the other output files. This was accounted for in the DDS toolkit but it makes analyzing the data substantially more bothersome as special rules had to be made when processing that data.

Problems II and III have been addressed and should no longer be present in the newest version of the DDS toolkit, although old data still has this problem. Problem I is much more difficult to solve in the DDS toolkit and would require some redesigning, but such changes would need to be implemented either way if the DDS toolkit is to be made to use a database, such as SQLite which was used during this project, instead of the old system of outputting everything to .txt files.

6.5.3 Summery of the problems with DDS toolkit

Based on the problems mentioned above, in particular the problem with super-directories, it's clear that the results regarding the duplicate files produced by the DDS toolkit can not be trusted and major changes have to be made to it before we can rerun the tools for the storage locations and acquire accurate data. Luckily better solutions have already been developed during this project, so we only need to implement them as part of the DDS toolkit.

6.6 Error with SQLite

A large part of the development time was dedicated to solving an error where the code would after it had found duplicate files, attempt to write to the table in the database from the previous iteration of the code. This is a known error with SQLite, and perhaps other SQL databases, where the system does not close the connection to the database after a query, and therefore old results may overwrite new values. This problem was discovered late in the project and after analysis it was determined that it could not be corrected for in post, meaning all the data had to be regenerated and reanalyzed, leading to a significant loss in time.

6.7 LDCS findings

Independent of this project the LDCS team was looking into a discrepancy between what was reported in a local job queue and the queue reported by ACT. This problem was discovered when they received warnings for quota violations while only having 175 jobs on the ACT queue. The actual batch quota for this is 300 jobs, meaning that the local job queue shows more than 300 jobs submitted while ACT only sees 175. David Cameron suggested that these are ghost jobs, meaning jobs that are running but are not tracked by ACT, and it was discussed that these jobs could be the source of the duplicate files problem. This theory does not seem unreasonable, as the fact ghost jobs would be untracked by ACT would explain the discrepancy between the number of jobs reported and the number of files outputted.

I was not aware that these findings were made, as I operated almost entirely on my own during this project, and knowing these findings I would have adapted to this new information. During this project, we only examined the data reported by the ACT, as I did not have access to the data from the local queue. Even before the submission of the project proposal, we did consider comparing the information about the number of jobs submitted to ACT with the number of jobs successfully registered by ACT. This was put aside as the majority of the project was conducted during July and August, when many of the LDCS members were on vacation, so we did not want to begin something that would require significant assistance from other LDCS developers. In retrospect, researching this should have been a greater priority and better use of my time compared to looking into the data from just ACT and the metadata from Rucio.

7 Conclusion

The duplicate file situation is a significantly bigger problem, particularly for the storage at Lund GridFTP, than what was predicted and would need to be properly addressed. During this project, we were able to generate evidence to disprove several prominent theories about the source of the duplicate files. We also formulated a new theory for what could be the source of this problem but said theory leaves many questions unanswered which would require additional research into.

The tools generated during this project could be used in the future to verify whether we have solved the duplicate file problem or not. While several problems were discovered with the DDS toolkit, we also were able to solve several such problems as well as implement many new features, such as multi-threading.

8 Acknowledgements

I want to thank the department of particle physics at Lund University for the funding and for allowing access to computational resources needed for the completion of this project. I would also like to thank my supervisor Ruth Pöttgen, Senior lecturer at Particle Physics at Lund University, as well as all the other members of the LDMX and the LDCS team for helping me along the way. In particular, I want to thank Lene Kristian Bryngemark, Ph.D. Department of Physics at Stanford University, for providing insight into the "test" data and the research done by the LDCS team about the mismatch between the local job queue and the ACT job queue.

9 Figures and tables

Metadata record for simulations

attribute name	type/format	example	source	description
SampleId	string	v9-8GeV-1e-ecalpn	config	The name of the sample the file belongs to
IsSimulation	yes/no	yes	Known by origin	Specifies whether the data file contains simulation data or detector data.
LdmxImage	string	ldmx-v1.7.0-gLDMX.10.2.3_v0.3-r6.18.04-el7-d7.sif	Determined from the RTE specified in the config	The name of the image file containing the LDMX software stack
ComputingElement	string	l-pilot.lunarc.lu.se	Comes from ACT	The name or ID of the computing cluster that generated the data
JobSubmissionTime	timestamp		Comes from ACT	The timestamp when the job was created in the production system
FileCreationTime	timestamp		cluster	The date/time when the simulation output file was created
Walltime	number	2358789	cluster	Walltime minutes used to generate the data
ARCCEJobID	id		cluster	The local ID of the simulation task within an ARC CE
BeamEnergy	number	8.0	Config and mac	Beam energy specified in GeV.
ElectronNumber	integer	1	Config and mac	Number of electrons in a bunch.
PhysicsProcess	string	ecal-photonuclear	Config and mac	The simulated physics process.
DetectorVersion	string	v9	Config file	Version of the detector design used in the simulation.
MagneticFieldmap	string	BmapCorrected3D_13k_unfold	Config file	The name of the file containing the magnetic field info

Figure 15: Metadata
An example of the metadata generated from a simulation and registered in Rucio for each output file.

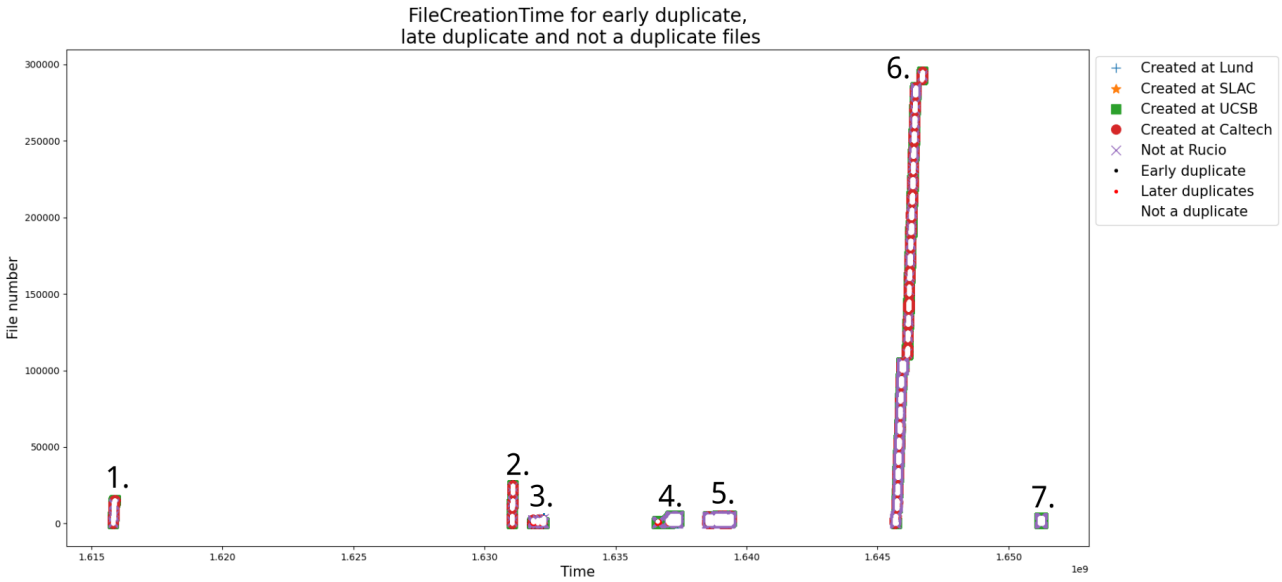


Figure 16: Complete plot over all files
A plot showing the FileCreationTime, which is the same as the file's timestamp, against the file number as explained in [2]. Files are marked based on what computational center they were created at, if the file is only present in storage and is missing from the Rucio catalog, and if the file is a duplicate. Among duplicates sharing the same FBT the duplicate with the earliest timestamp is marked as an "Early duplicate" and all duplicates with a later timestamp are marked "Later duplicates"

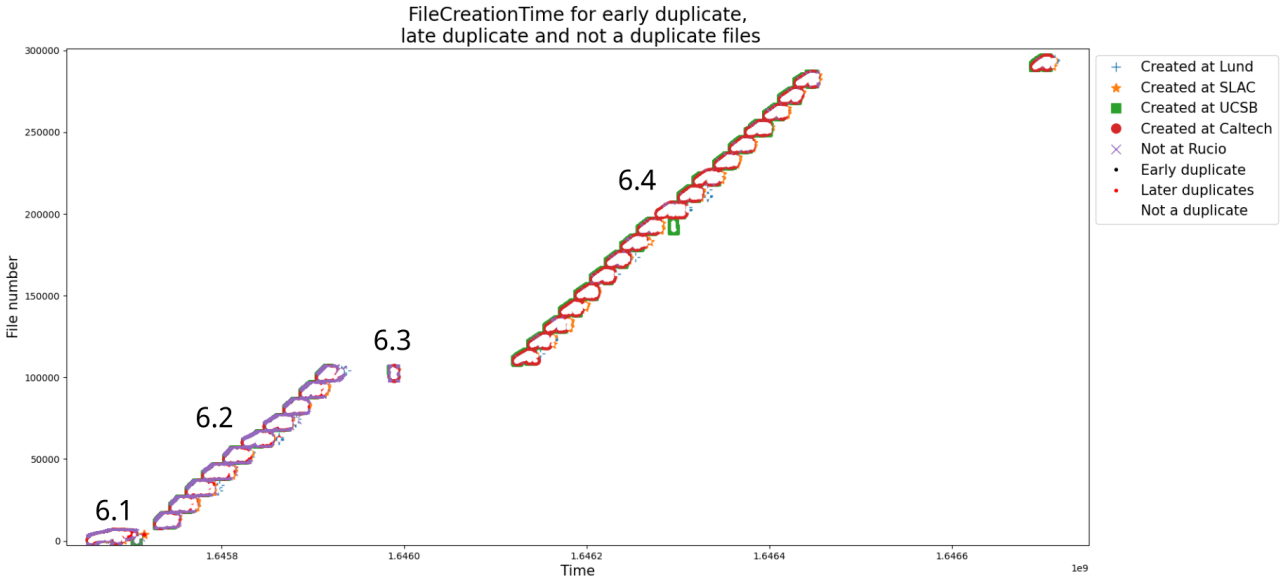


Figure 17: A plot showing a subsection of the complete plot [16] zoomed in on the cluster denoted as 6.

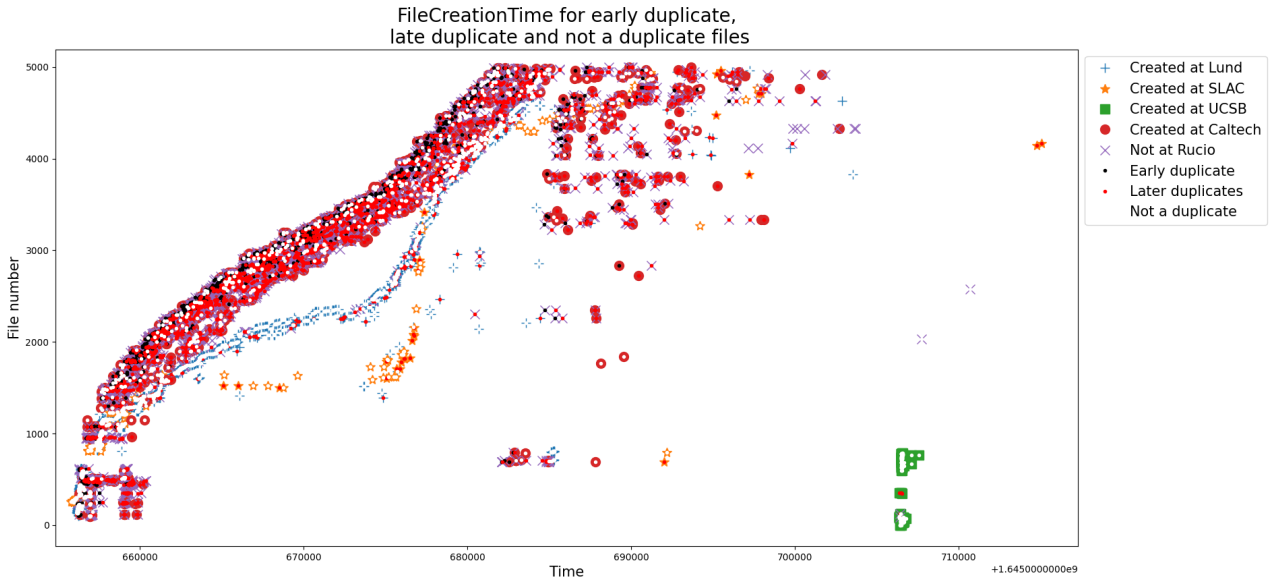


Figure 18: A plot showing a subsection of cluster 6 [16] zoomed in on the sub-cluster denoted as 6.1.

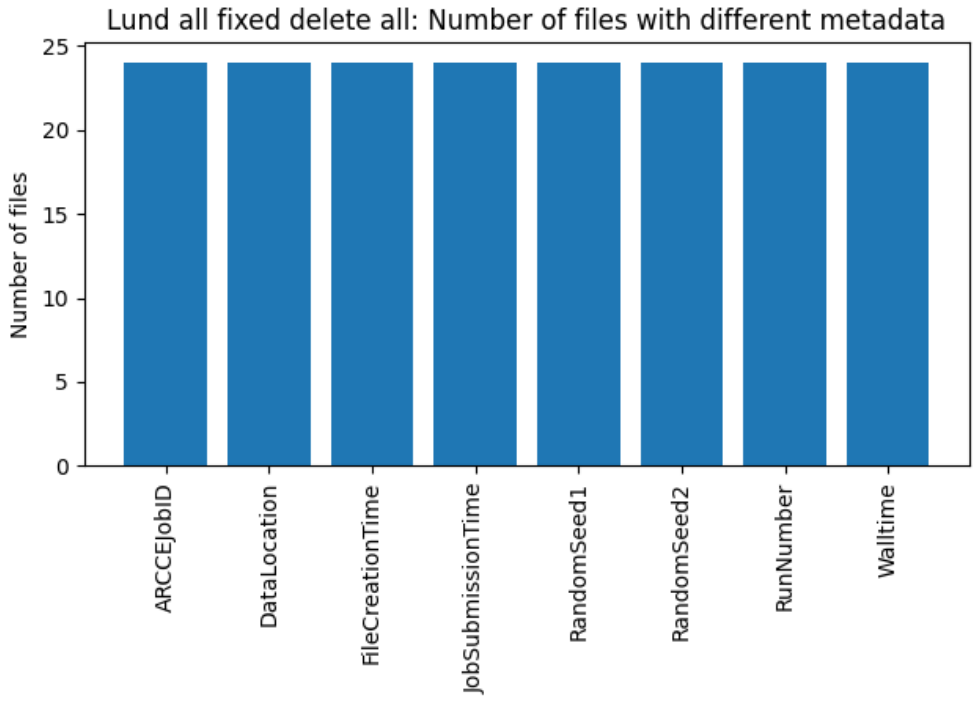


Figure 19: Metadata Lund
A bar plot showing what metadata attributes differ between all the files in a dataset that contains at least one duplicate file for storage at Lund. The y-axis displays the occurrence of such differences and on the x-axis, we have the different attributes.

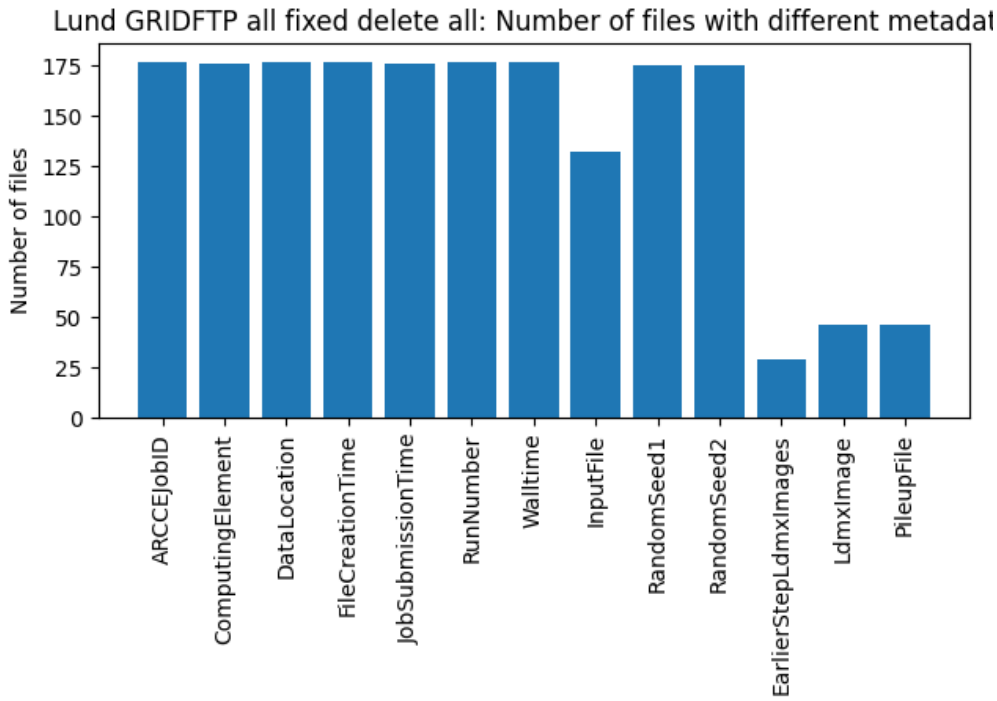


Figure 20: Metadata Lund GridFTP
A bar plot showing what metadata attributes differ between all the files in a dataset that contains at least one duplicate file for storage at Lund GridFTP. The y-axis displays the occurrence of such differences and on the x-axis, we have the different attributes.

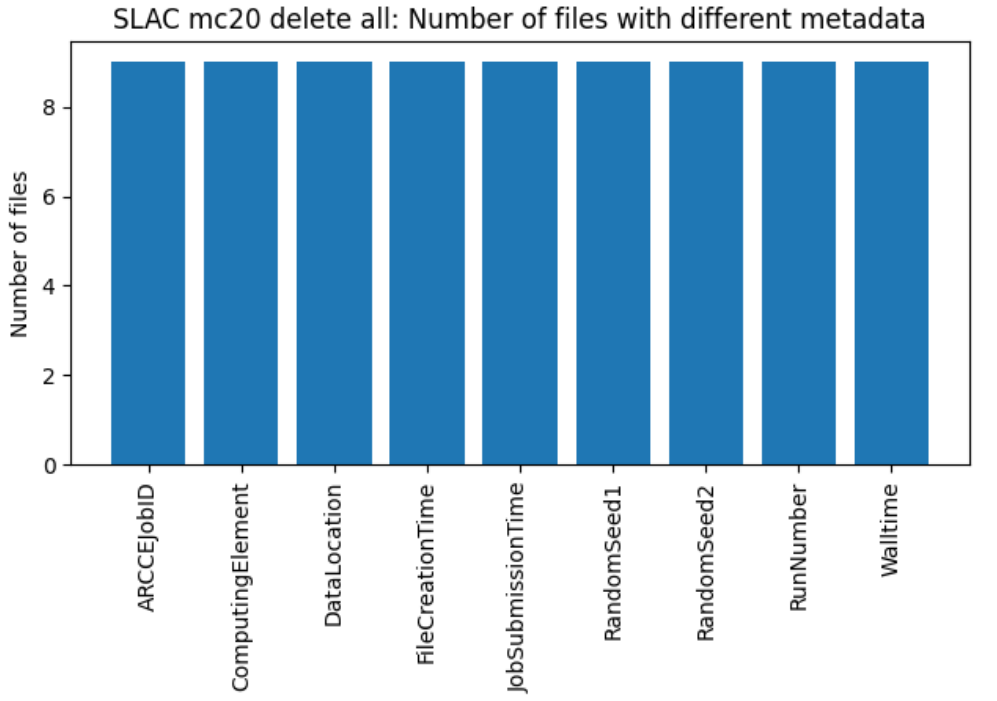


Figure 21: Metadata SLACbar plot showing what metadata attributes differ between all the files in a dataset that contains at least one duplicate file for storage at SLAC. The y-axis displays the occurrence of such differences and on the x-axis, we have the different attributes.

10 Method for generating the database

10.1 Extracting data

To analyze the metadata for the duplicate files it has to be retrieved from Rucio and organized. To receive the metadata from the Rucio catalog we need to provide the function:

```
rucio get-metadata scope:filename.root
```

with the scope to which the file belongs, which is a large dataset containing smaller datasets, and the filename itself. In the first version of the **get_metadata.py** script, this information had to be provided manually by the user for each dataset or extracted from the name of the directory. The scope could often be determined from the name of the directory storing the file. As an example, from the directory

/projects/hep/fs9/shared/ldmx/ldcs/output/ldmx/mc-data/**mc20**/v12/4.0GeV/v2.2.1-1e

we can conclude that the scope was mc20. This is not a consistent method of extracting the information about the scope because situations arise where the directory name does not contain the scope. While this method did not always work, it was still good enough for a manual limited test.

Later versions would use the output data that we generated by running the DDS toolkit, which we designed as part of the thesis project, as input. Two output files are of interest to us from the DDS toolkit:

files_found_storage.txt

A text file that contains a list of all the files that the system found to exist both in storage and in the Rucio catalog.

files_missing_rucio.txt

A text file that contains a list of all files that only exist in storage, and are not registered in Rucio.

We need to use the information from both of these lists, as we want to observe when we have one duplicate file registered in Rucio and one, or more, duplicate files missing from the Rucio catalog. The output from the original version of the DDS toolkit did not state the scope of the files found in both storage and Rucio. For the storage at Lund with and without GridFTP access, to which we have direct access, we modified the original DDS toolkit and ran the modified tool to generate new output files with the information that was previously missing. For the storage at SLAC, we did not have direct access to the storage during this project so we could not simply rerun the new improved version of the DDS toolkit. Luckily we only ran the tool for the scope mc20 during the thesis project, so the scope could manually be set to mc20 for all the files at that storage location.

We then ran a script that would create an SQLite database with a table for each dataset. That table would be filled with these values for each file entry:

id: An integer that goes from 1 to the number of files in a dataset.

file: The filename

BatchID: The name of the dataset the file belongs to.

ComputingElement: The name of the computing element where the file was produced

DataLocation: The name of the data storage facility where the file is currently stored

Scope: What scope does the file belong to.

JobSubmissionTime: At what time was the job submitted to a computation site, written as

```
YEAR-MONTH-DAY HOUR:MINUTE:SECOND
```

FileCreationTime: At what time was the output file from the simulation created, measured in UNIX-time

IsRecon: Is set to True if the job was a reconstruction and False if it was a simulation job.

The files that we wished to check were put into a list, and that list was split into 8, and in a later version 16, equal parts. We then created 8 or 16 threads and ran them for the files in the different lists.

If the Rucio server returned an error saying the file was not found in the Rucio catalog, the values in the table were set to:

id=the position in a list of all the values

file=The filename provided to the function

BatchID=None

ComputingElement=None

DataLocation=None

Scope=None

JobSubmissionTime=None

FileCreationTime=The timestamp in the filename, which always matched the value I got from the Rucio

IsRecon=None

10.2 Pre-processing the data

Before the data in the SQLite database could be analyzed there reminded some information that could be extracted.

10.2.1 Reformating the timestamp

The JobSubmissionTime value was written in the form

`YEAR-MONTH-DAY HOUR:MINUTE:SECOND`

which did not match the UNIX time used by the FileCreationTime, so the date-time format was changed to the equivalent UNIX timestamp.

10.2.2 Add file number

Each filename contains a number that is supposed to be unique for every file in a simulation job but is shared among duplicate files.

$$\underbrace{\underbrace{\text{mc\_v9-8GeV-1e-target\_photonuclear}}_{\text{Unique for every simulation run}}\_\overbrace{14569}^{\substack{\text{Unique number for each}\\ \text{file in simulation run}}}\_\underbrace{t1589280908}_{\text{timestamp}}.root}_{\text{Unique for every file}} \quad (2)$$

This number is extracted from the file name and is added as a separate column in the table. The part of the filename before the timestamp we call Filename Before Timestamp (FBT).

10.2.3 Finding all duplicates

Using the file number extracted in the last step, we iterate over all the file numbers and find if any files share a number. Every set of files with the same file number is then sorted by the FileCreationTime. The file with the earliest timestamp is set to be duplicate number 1, the second earliest duplicate to be number 2, etc. These numbers are then written to a new column called **duplicate**. If a file number only has one file associated with it, meaning that the file has no duplicates, the value in the duplicate column is set to None.

11 References

[1] AAkesson, Torsten, Asher Berlin, Nikita Blinov, Owen Colegrove, Giulia Collura, Valentina Dutta, B. Echenard, et al. The Light Dark Matter EXperiment (LDMX). Proceedings of The 39th International Conference on High Energy Physics PoS(ICHEP2018), 2019.

[2] Raubenheimer, Tor, Anthony Beukers, Alan Fry, Carsten Hast, Thomas Markiewicz, Yuri Nosochkov, Nan Phinney, Philip Schuster, and Natalia Toro. DASEL: Dark Sector Experiments at LCLS-II. arXiv, 2018. https://-doi.org/10.48550/ARXIV.1801.07867.

[3] Bryngemark, Lene Kristian, David Cameron, Valentina Dutta, Thomas Eichlersmith, Balazs Konya, Omar Moreno, Geoffrey Mullier, et al. Building a Distributed Computing System for LDMX. EPJ Web Conf. 251 (2021). https://-doi.org/10.1051/epjconf/202125102038.

[4] Yartsev, Piotr, Improvement of the Rucio Implementation for the LDCS Platform and Search for Dark Data, 2022.