Computing and Software for Big Science (2025) 9:15
https://doi.org/10.1007/s41781-025-00144-5

RESEARCH

The LHCb Sprucing and Analysis Productions

Ahmed Abdelmotteleb[1] · Alessandro Bertolin[2] · Chris Burr[3] · Ben Couturier[3] · Ellinor Eckstein[4] · Davide Fazzini[5] · Nathan Grieser[6] · Christophe Haen[3] · Ryunosuke O'Neil[3] · Eduardo Rodrigues[7] · Nicole Skidmore[1] · Mark Smith[8] · Aidan R. Wiederhold[9] · Shunan Zhang[10]

Received: 24 June 2025 / Accepted: 29 July 2025

Abstract

The LHCb detector underwent a comprehensive upgrade in preparation for the third data-taking run of the Large Hadron Collider (LHC), known as LHCb Upgrade I. With its increased data rate, Run 3 introduced considerable challenges in both data acquisition (online) and data processing and analysis (offline). The offline processing and analysis model was upgraded to handle the factor 30 increase in data volume and the associated demands of ever-growing datasets for analysis, led by the LHCb Data Processing and Analysis (DPA) project. This paper documents the LHCb "Sprucing" — the centralised offline data processing and selections — and "Analysis Productions" — the centralised and highly automated declarative nTuple production system. The DaVinci application used by analysis productions for tupling spruced data is described as well as the `apd` and `lbconda` tools for data retrieval and analysis environment configuration. These tools allow for greatly improved analyst workflows and analysis preservation. Finally, the approach to data processing and analysis in the High-Luminosity Large Hadron Collider (HL-LHC) era — LHCb Upgrade II — is discussed.

Keywords Data-processing · Computing · LHC

✉ Chris Burr
christopher.burr@cern.ch

✉ Nicole Skidmore
nicola.skidmore@cern.ch

Ahmed Abdelmotteleb
ahmed.abdelmotteleb@cern.ch

Alessandro Bertolin
alessandro.bertolin@pd.infn.it

Ben Couturier
Ben.Couturier@cern.ch

Ellinor Eckstein
ellinor.eckstein@cern.ch

Davide Fazzini
davide.fazzini@cern.ch

Nathan Grieser
nathan.allen.grieser@cern.ch

Christophe Haen
christophe.denis.haen@cern.ch

Ryunosuke O'Neil
r.oneil@cern.ch

Eduardo Rodrigues
Eduardo.Rodrigues@cern.ch

Mark Smith
mark.smith@cern.ch

Aidan R. Wiederhold
aidan.richard.wiederhold@cern.ch

Shunan Zhang
shunan.zhang@cern.ch

[1] Department of Physics, University of Warwick, Coventry, UK

[2] INFN Sezione di Padova, Padova, Italy

[3] European Organization for Nuclear Research (CERN), Geneva, Switzerland

[4] Universität Bonn - Helmholtz-Institut für Strahlen und Kernphysik, Bonn, Germany

[5] INFN Sezione di Milano-Bicocca, Milano, Italy

[6] University of Cincinnati, Cincinnati, USA

[7] Oliver Lodge Laboratory, University of Liverpool, Liverpool, UK

[8] Imperial College London, London, UK

[9] Department of Physics and Astronomy, University of Manchester, Manchester, UK

[10] Department of Physics, University of Oxford, Oxford, UK

Published online: 04 August 2025

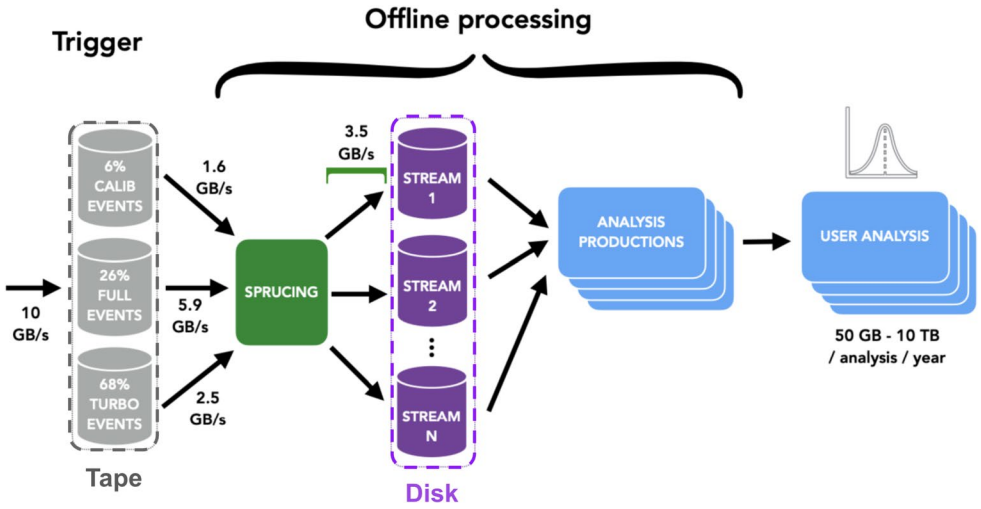

Fig. 1 Schematic of the LHCb offline data flow. Figure adapted from Ref. [5]

Introduction

The LHCb experiment is one of the four main experiments collecting data from proton-proton collisions at the Large Hadron Collider (LHC) [1]. It is a forward arm spectrometer specialising in the decays of beauty and charm hadrons. During the first and second LHC data collection runs — Runs 1 & 2 — LHCb collected data corresponding to an integrated luminosity of 9fb^{-1}, equating to over 10^{12} $b\bar{b}$ pairs in the acceptance of the detector. In 2022 the LHC commenced its third run of data-taking known as Run 3. For Run 3 LHCb underwent a comprehensive upgrade — known as LHCb Upgrade I — in anticipation of a factor 5 increase in delivered luminosity [2]. This equated to an increase of more than a factor 30 in the volume of data collected by LHCb per unit of time, taking into account the increase in delivered instantaneous luminosity, a factor 3 due to the increased average event size, and a factor 2 due to higher trigger efficiencies [3]. Run 3 therefore posed not only data collection (online) challenges, but also significant offline data processing and analysis ones. This paper documents the two key developments to facing these challenges:

Sprucing Centralised offline data skimming, slimming and splitting of the data into multiple data streams to reduce the data footprint between tape and disk storage.
Analysis productions Centralised `nTuples` production using the LHCb DIRAC [4] transformation system with maximal automation and optimised user experience.

The LHCb Upgrade I offline data and processing flow is sketched in Fig. 1, showing the role of the Sprucing and Analysis Productions.

Sprucing

The Role of the Sprucing in LHCb Upgrade I

The LHCb High-Level Trigger (HLT) comprises two stages — HLT1 and HLT2. The LHCb online system writes events that pass the HLT2 to tape in three physics streams, discussed below: `FULL`, `TURBO`, and `TURCAL`.[1] A physics event will populate one of these streams if it passes a HLT2 selection line that belongs to the stream. In Run 3 LHCb has a design data rate to tape of 10GB/s. The primary purpose of the Sprucing is to split the three physics streams into sub-streams and reduce the total data rate to 3.5GB/s, which was determined a priori to be sustainable for offline disk resources throughout Run 3. How the Sprucing achieves this rate reduction is stream dependent, as described in the following section; Fig. 2 illustrates the methods used for each physics stream.

The LHCb online processing produces and exports data in the Mast Data Files `MDF` format [6]. In `MDF` file format, each event is stored sequentially such that files can be merged by concatenation. Problematic files can be partially recovered by starting the processing from the first valid event. Although the `MDF` format provides many convenient features for the LHCb data acquisition system, it is limited to only storing per-event data. Furthermore, `MDF` files tend to be large in size due to the lack of inter-event compression. To mitigate the effect of large file sizes, `MDF` files are compressed with `Zstandard` [7] when exported from the LHCb detector site to World-wide LHC Computing Grid (WLCG) [8] resources.

The `MDF` files exported from LHCb contain so-called *RawEvents*, each of which consist of a collection of

[1] There are an additional three technical streams that are written.

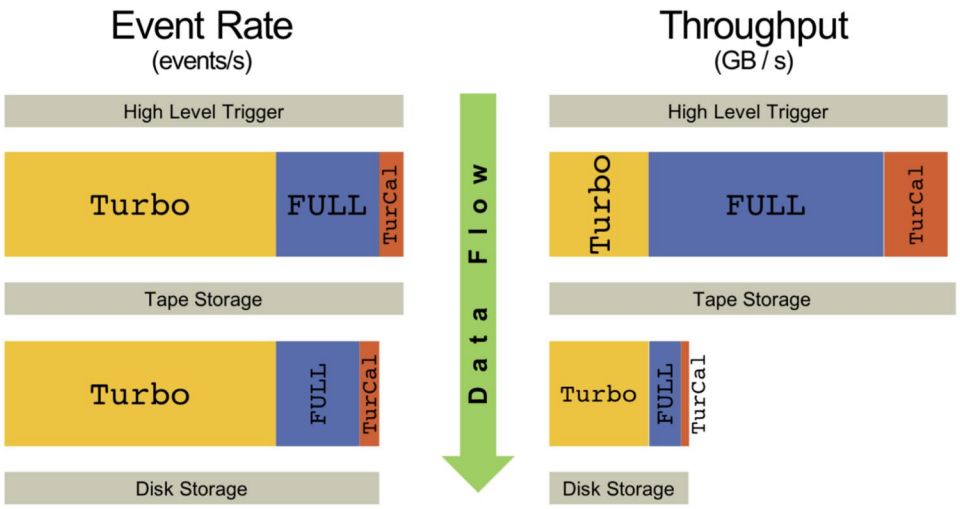

Fig. 2 The Sprucing reductions in event rate and throughput for the FULL, TURBO and TURCAL streams. Left: event rates. Right: throughput in GB/s. Box widths are proportional to the corresponding quantities. Figure taken from Ref. [3]

RawBanks containing different types of event data. A RawBank can store:

- Physics candidate information such as the 4-momenta of particles in a decay-chain and any reconstruction objects requested. This is known as the "DstData"RawBank.
- HLT{1,2} or Sprucing line decisions. These are called the "HltDecReports"RawBanks.
- Detector response information - eg. from the calorimeter system (CALO) or muon chambers (MUON).

The Sprucing step outputs so-called DST files in the ROOT file format, which again consist of, but are not limited to, RawEvents consisting of RawBanks. These DST files are distributed to WLCG sites and saved on disk to be available to analysts.

The FULL stream

The FULL stream contains inclusive HLT2 selection lines such as the topological lines. The advantage of the FULL stream model is that the full event reconstruction is persisted to tape and so this data can be re-processed with new or updated selection lines periodically many years into the future as discussed in Sect. Re-Sprucing Campaigns. This enables LHCb 's legacy physics programme, exploiting the LHCb dataset for new physics channels long after LHCb stops taking data.

For the FULL stream the Sprucing slims and skims events by running additional, exclusive selection lines on top of these inclusive events offline. The Sprucing runs selections using the same application — the Moore application [9] — as the online system, and hence HLT2 selection lines and Sprucing selection lines are identical and, by design, trivially interchangeable. Furthermore, the same algorithms and tools are shared between HLT2, Sprucing and the offline analysis software project DaVinci [10], namely the ThOr [11] based selection and combinatorial algorithms. This was a conscious choice and departure from the Run 2 model. Compared to HLT2, the Sprucing selection lines benefit from less strict limits on the timing of selection algorithms. This makes the FULL stream particularly important for physics channels with many final-state particles, which may suffer from large combinatorial backgrounds. Lines such as $B^0 \to D^* hhh$ with $D^* \to D^0\pi$ and $D^0 \to hhhh$ where $h = K, \pi$, unavoidably require the computation of a large number of particle combinations to build vertices and composite particles that can be selected. The consequent reduction in throughput means that these lines cannot be run at the HLT2 stage and must instead be run by the Sprucing on the data saved by the inclusive lines of the FULL stream.

The Sprucing, via exclusive selection lines, is required to achieve a factor 7 reduction in the FULL stream bandwidth so that this data can be saved to disk. For efficient data access by analysts, the Sprucing is tasked with further splitting the data into sub-streams and creates $\mathcal{O}$ (20) DST outputs based on physics case and line rate. The Sprucing step also creates File Summary Records (FSRs) that record luminosity information useful for offline analysis, meaning that the luminosity events can be discarded at this stage. FSRs are described in more detail in Subsection File Summary Records (FSRs).

At the end of the 2024 data taking period, the FULL stream contained 401 (mainly) inclusive HLT2 lines, and the Sprucing subsequently ran 1138 Sprucing lines on this stream with the output saved to disk.

The TURBO stream

In the TURBO stream [12], HLT2 selection lines are generally exclusive to a particular decay channel and persist only a custom set of physics objects and their reconstruction,

minimally the triggering candidate and primary vertices in an event. This became the Run 3 default model, and to achieve the baseline of 10GB/s to tape, LHCb required 73% of its physics programme to use TURBO. In general, lines that select well-known simple decay topologies, such as $B^0 \to D^-\pi^+$, are included in the TURBO stream. With relatively simple selections, it is possible to achieve sufficiently small rates, consistent with the overall requirement of a 2.5 GB/s bandwidth to disk. It should be noted that the TURBO model is highly flexible and any object including raw detector banks can be selectively persisted. In the TURBO stream, events are already exclusively selected for a dedicated decay channel with only the necessary event information persisted for that decay. Therefore, no further slimming of this data is required by the Sprucing. The Sprucing writes the TURBO data to $\mathcal{O}$ (20) sub-streams and populates the FSRs. At the end of the 2024 data taking period, the TURBO stream was running 2502 lines, which were subsequently spruced.

The TURCAL stream

Finally, the TURCAL stream contains selection lines for physics channels that are used for, among other studies, detector calibration. Hence these lines additionally persist raw detector information (CALO or MUON for instance). In the case of TURCAL, a factor 8 reduction is again required by the Sprucing and the data is further streamed. The TURCAL Sprucing achieves this by facilitating a line-by-line customisation of the persistency of reconstruction objects and detector RawBanks to reduce the data footprint to disk. The TURCAL stream, like the FULL stream, maintains the option for re-processings and even re-running the reconstruction due to the presence of raw detector information. At the end of the 2024 data taking period, the TURCAL stream was running 147 lines, which were subsequently Spruced.

Sprucing Productions and Campaigns

The Sprucing runs in centrally-managed offline productions, using the LHCb DIRAC transformation system to achieve massive parallelism by distributing single-threaded jobs across the WLCG. Generally, the Sprucing runs on WLCG Tier-0 and Tier-1 sites as these sites host the tape systems that store the input data. This multiprocessing approach is used as Sprucing's performance is limited by I/O and scheduling overhead, not CPU-intensive calculations. Consequently, running multiple single-threaded jobs is just as efficient and avoids the potential performance penalties associated with thread management.

Productions are created through YAML files and tracked through GitLab [13, 14] issues by Sprucing coordinators and production managers. Continuous Integration is used for selection line development — performed by hundreds of collaboration members — and maintenance with automated rate and bandwidth tests performed on a nightly basis or triggered on demand [15]. Whenever possible, these tests use ad-hoc recorded data samples as input. This ensures a fully tested suite of selection lines at all times and avoids the line preparation campaigns of Runs 1 & 2 that lasted several weeks.

Concurrent Sprucing Campaigns

Similarly to Stripping of Runs 1 & 2 [16], the Sprucing runs concurrently with data taking. Multiple Sprucing campaigns are established over a data-taking year corresponding to changes in the configuration of HLT1, HLT2 or the Sprucing itself that impacts the physics output, for example, updates to existing selection lines or the addition of new ones. The development schedule of trigger and Sprucing campaigns is closely aligned and coordinated.

Sprucing campaigns are validated efficiently via the use of Analysis Productions (described in detail in Sect. Analysis Productions). nTuples are produced from a small subset of Spruced files that analysts can verify interactively via the Analysis Productions web browser integrated with tools such as JavaScriptROOT [17, 18].

In nominal, concurrent campaigns, the observed turnaround between the HLT2 processing, file transfer to the Offline system, and the Sprucing processing is 24-48 h.

Re-Sprucing Campaigns

Re-Sprucing campaigns take place in end-of-year LHC shutdowns when the Sprucing input data can be staged from tape to disk and Offline resources have capacity. This (typically) four-month period constitutes a strict time frame within which the re-Sprucing must be completed to avoid conflict with concurrent campaigns.

Given the full event reconstruction persisted in the FULL stream, re-Sprucings offer the opportunity to re-optimise selections or add new selection lines based on the needs of physics analysts. The preparation of re-Sprucing campaigns starts several weeks before the end of data-taking of the year, when analysts submit requests and implement changes to selection lines.

Once the concurrent data processing has finished, the Sprucing input data is staged from tape to disk. A small fraction of data is re-Spruced as part of the campaign validation with Analysis Productions, again, used by analysts to check distributions of variables. Once the green-light is given, the full production is launched in order to complete the processing before the start of the next year's data-taking.

Re-Sprucing campaigns can either be *full* — whereby all Sprucing lines are rerun and the concurrent campaign is superseded and can be removed from disk — or *incremental*

— where only new or modified lines are run and this dataset is additional to the previous *full* (re-)Sprucing dataset. The decision between running an *incremental* or *full* campaign is made based on the number of lines added/modified and the resulting storage requirements.

Heavy Ion Data Runs

LHCb not only takes event data from proton-proton (*pp*) collisions, but also heavy-ion collisions provided by the LHC. This data is also Spruced. Two physics streams are output from the HLT2: `ION` and `IONRAW`. The `IONRAW` stream undergoes a simple pass through in the Sprucing, similar to *pp* `TURBO` data. However, the `ION` stream undergoes a similar processing as the *pp* `TURCAL` stream where, line by line, the persistency of reconstruction objects and detector RawBanks can be customised, reducing the data footprint to disk. This data can be re-Spruced in the future should the extra persistency objects that are on tape be required.

File Summary Records (FSRs)

A File Summary Record (FSR) is a per-file tree data structure containing information on the `DST` file content, stored within the `DST` file itself. FSRs cannot be used in the LHCb `MDF` file format due to their "per-event"storage as described in Sect. The Role of the Sprucing in LHCb Upgrade I, so Sprucing is the first point in the data flow in which FSRs can be created. FSRs have been used since LHCb Run 1 to record luminosity information. The total number of luminosity events in a file, identified by a unique routing bit, is recorded inside the FSR to indicate the LHCb luminosity a file represents. Storing this information in an FSR means that these events can be discarded at the Sprucing stage.

In Run 3 the FSR information was expanded to include the decoding tables and the application options of the last processing step making Spruced files fully self-contained. As described above, LHCb data contains a "DstData"RawBank for physics candidate information and a "HltDecReports"RawBank for the trigger decisions of each event. To save disk space, this data is encoded. The (potentially long) strings of physics object locations (`PackedObjectLocations`) and the selection line decision names (`HLT1SelectionID`, `HLT2SelectionID`, `SpruceSelectionID`) are mapped to integer values. In the RawBanks, only the integers and corresponding information are stored per event and so, in order to read the data, these integer-to-string mappings, referred to as decoding tables, are necessary. Each decoding table is identified by a unique hexadecimal decoding key and stored. To make the Sprucing output files self-contained, the decoding tables are now written to the FSR and can be read from there in further processing steps.

Table 1 Stream size before and after Sprucing for data taking in 2024

Stream	Stream size before Sprucing (PB)	Stream size after Sprucing (PB)
`FULL`	3.08	0.45
`TURBO`	1.25	1.12
`TURCAL`	0.86	0.11

The stream size after Sprucing is calculated as the sum over all the sub-streams created from the stream by the Sprucing

Table 2 Average event size before and after Sprucing for data taking in 2024

Stream	Av. evt size before Sprucing (kB)	Av. evt size after Sprucing (kB)
`FULL`	114.9	33.8
`TURBO`	13.4	10.2
`TURCAL`	176.9	21.2

The application options used to create a Spruced file are also now stored in the file's FSR — these are the Moore options used to run the Sprucing job. Besides the provenance aspect of knowing exactly how a file was produced via metadata stored in the file itself, storing these options allows further automation of Analysis Productions (described in Sect. Analysis Productions), as well as configuration consistency checks, as configuration options to run the DaVinci application on Spruced files can be deduced from the FSR. These configuration options include the detector geometry version and data-taking conditions, the file type, the input process (*e.g.*, `HLT2`) and whether the file contains real or simulated data.

Sprucing Performance in Run 3

In 2024 LHCb recorded a record 9.56fb^{-1} of data as shown in Fig. 3. Bandwidth reductions of a factor 7 and 8 were achieved for the `FULL` and `TURCAL` streams, respectively. Note that there is also a reduction in the size of the `TURBO` stream due to the better compression available with the `DST` format. The stream sizes before and after the Sprucing can be seen in Table 1 with the corresponding average event sizes in Table 2. Due to the efficiency of the concurrent Sprucing productions and subsequent Analysis Productions (as described in section 3) analysts were able to begin analysing data within 2–3 days of it being recorded — a first for any LHC experiment and complementing the online Real-Time Analysis (RTA) ethos.

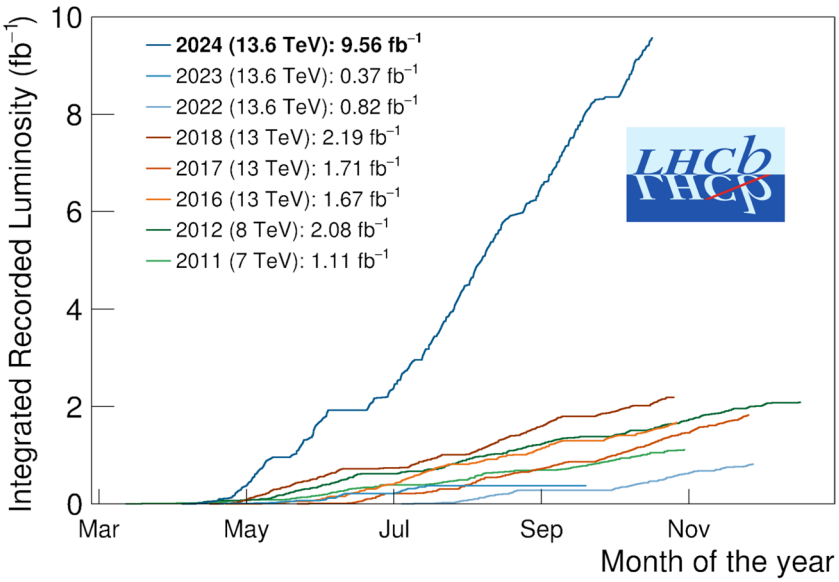

Fig. 3 Integrated luminosity recorded by LHCb per data-taking year from 2011 to 2024. Figure taken from Ref. [19]

Analysis Productions

The Role of Analysis Productions in LHCb Upgrade I

After the Sprucing procedure, the data is split into multiple streams (Fig. 1), which are distributed to WLCG sites and made directly accessible to analysts. In the Run 1 &2 analysis model, individual analysts would submit user jobs to LHCb DIRAC to create `nTuples` from these files. The analyst would have to monitor the jobs, handle issues in the workflow in case of failure, and manage the storing of the output. Although this pragmatic approach (the "user job model") was practicable for the Run 1 and/or 2 dataset, the approach does not scale for the following reasons:

Error recovery: With larger input data samples comes higher occurrences of faults in data processing or grid operations. In many cases, these require expert attention to recover from, for which the end-user is typically not equipped. Additionally, issues are often common among many analysts. In the "user-job"model, more work is done to fix these issues as it is not addressed through a single, central solution.
Human error: The cost (in resources and storage) of configuration errors increases with significantly larger input and job volumes.
Operational burden: Problematic user workflows can interfere with the wider operation of the Grid such as centralised data or simulation productions.
Data lifecycle: The lifecycle of the output data is not systematically managed. This data forms the basis of the analysis results and scientific output, and in the user job model the relevant provenance information is highly prone to loss.

Analysis Productions (APs) are an extension of the LHCb DIRAC transformation system that enables a simplified and declarative approach to the configuration of distributed computing workflows with LHCb applications. The majority use case in Run 3 is the transformation of the LHCb dataset into `nTuples`, rectangular datasets used for physics analysis, which are produced by the LHCb application DAVINCI described in more detail in Sect. he DAVINCI Application. APs provide a simplified user interface to configuring and testing workflows on the WLCG's vast and complex resources and constitute an essential and transformative development for LHCb in Run 3.

Declarative Tupling and the User Interface

Analysis Productions may be configured to run released LHCb applications or scripts, where each application or script (a step) transforms the input data into one or more output files. Steps can be chained together to transform the LHCb data in a variety of ways, but generally the first step for analysts is to run the offline analysis application DAVINCI to create `nTuples`.

Analysis Productions are defined declaratively via YAML files by describing the workflow, job configurations, and bookkeeping query for the input data, the latter enabling LHCb DIRAC to automatically adjust the way in which files are grouped and handle failures. Information about productions and the provenance of files is permanently stored in the so-called *LHCb bookkeeping system* catalog [20]. enabling high quality analysis preservation and additional safety checks to be performed, as well as efficient dataset cleanup and archival.

Analysis Productions are submitted by users via merge requests to a GitLab repository. Minimally, this merge

request contains LHCb application scripts — in most cases a DaVinci script — and the YAML describing the job. The AP repository is linked with LHCb Dirac through custom services that handle production testing and submission.

Production Testing and Submission

To validate user-prepared configurations and avoid wasted computing resources and instability in grid operations, extensive continuous integration tests are administered and supported by custom services (an API, celery task queue, and database) linked with LHCb Dirac. Tests are triggered by updates to the merge request. Submission of real productions are gated by the result of these tests and the approval from physics working group liaisons and, in some cases, LHCb grid experts. The validation process includes:

Job stability: Running the entire chain using a small (configurable) subset of input data.
Memory footprint: Detailed analysis of the application memory consumption versus run time using the programme `prmon` [21].
Expected storage footprintt: Estimation of the final `nTuple` sample size extrapolating from the validation results.

Should any monitored criteria exceed predefined thresholds, the tests will fail, and suggestions to fix the problem are provided to the user (via the merge request) on the AP GitLab platform. The productions, their status, and details of the validation are summarised on the platform, connected to aforementioned services, with curated job logs highlighting potential warnings and errors.

Successful productions are run on behalf of the user. Production configurations and the provenance of files are permanently tracked in the LHCb Dirac Bookkeeping database. Users can tag multiple analysis productions indicating that they belong to the same measurement; these tags are used by the `apd` [22] tool to allow easy data retrieval as described in Sect. The `apd` Tool and Analysis Preservation. This enables high-quality analysis preservation, as well as efficient data management and archival. The storage usage of `nTuples` is closely monitored, split by both the physics working group and individual analyses using `apd` tags.

Analysis Productions Performance in Run 3

At the time of writing, over 2000 analysis productions have been submitted by analysts on data and simulation. Analysis Productions can be setup to remain “live"such that they automatically process the data as it arrives out of concurrent Sprucing productions; this means analysts can access their data `nTuples` within days of the data being recorded. Analysis Productions resource uptake follows that of the Sprucing and at its peak in 2024 Analysis Productions were able to process 14PB of data in a single day as shown in Fig. 4.

The `apd` Tool and Analysis Preservation

The `apd` tool provides a simple programming interface to the Analysis Productions system. It is a Python package published to standard repositories (such as `PyPI` [23] and `conda-forge` [24]) that allows the look-up of the physical file names of files created in the context of an Analysis Production, filtering them according to associated metadata (tags). Some of these tags are automatically created by the system (*e.g.*, the data type), but analysts can also define custom tags that indicate the dataset properties or its intended use. Using `apd`, analysts can avoid keeping lists of physical file names that have time-limited validity and instead declare input data with information that represents its properties and intended use (*e.g.*, the analysis name, the working group it is attached to, the data-taking year, the event type in case of simulated data, etc.).

Calls to `apd` within an analysis code base allow clear access to the provenance of the data. To further improve this, an interface to the workflow management system `Snakemake` [25] was also designed and implemented; `Snakemake` has seen significant adoption in LHCb analyses. The integration of `apd` into `Snakemake` allows tracking of

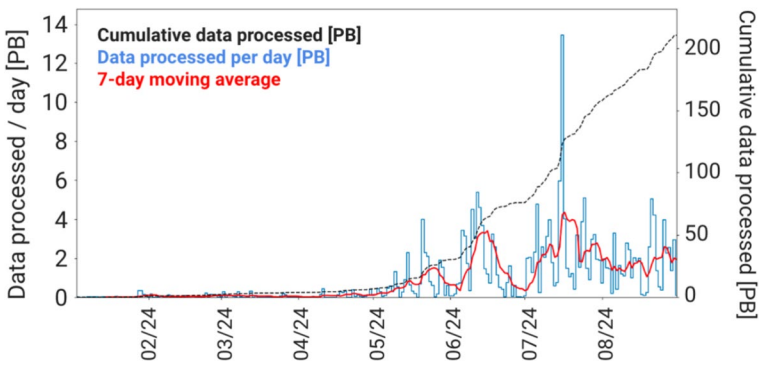

Fig. 4 Data processed per day and cumulative data processed by Analysis Productions in 2024

dependencies between the various data artifacts produced by the analysis up to the original files created by the Analysis Production.

The ease of use is crucial to help with the adoption of any new system. The `apd` tool is installed in the default LHCb analysis environment which is available through `lb-conda`. The `lb-conda` tool defines LHCb Conda wrapper scripts [26] that provide access to Conda versions on CERN's `cvmfs` [27]. The `apd` tool has few dependencies and can be installed on any machine independently of the other LHCb software applications if necessary. Using `apd` from Python scripts requires minimal effort from analysts, and also provides the functionality to cache metadata and files locally, to improve performance, or reproduce part of the analysis locally.

Historically LHCb has seen issues with the preservation of analyses due to the evolving nature of computing. This means that even if the analysis and software is well preserved, the ability to connect to external systems to access the data eventually stops working. This can be due to security developments — for example, the deprecation of the `TLS 1.0` security protocol [28] or Certificate Authority changes — or modernisation of storage systems such as the migration to `EOS` [29] at CERN. The `apd` tool solves this by abstracting the network connectivity through a common interface meaning, in the long term, it is then possible to substitute the service `apd` uses to lookup files with a local `HTTP` [30] server or even file URLs. Data access with `apd` nominally uses `XRootD` [31] however this can be transparently changed to return paths to copies on local `POSIX` storage.

Combining the use of `apd` to search for LHCb Analysis Production data files, `lb-conda` to ensure the reproducibility of the analysis environments and `Snakemake` to define the analysis workflows ensures comprehensive and reliable analysis preservation. The ease of use of this system has led to significant uptake from LHCb analysts.

The DaVinci Application

Analysis Productions uses the offline analysis software project DaVinci to create `nTuples` for further high-level analysis. The DaVinci application is built on the Gaudi [32] framework software package for processing High Energy Physics event data. In Run 3 DaVinci employs the purpose-built `FunTuple` component to facilitate the storage of event data in `ROOT` format, optimising it for subsequent offline analysis [33]. These `nTuples` may contain information about the selected signal candidates and the decay products such as particle kinematics, particle identification hypotheses, vertex fit qualities, etc. Within DaVinci one can run the DecayTreeFitter algorithm [34] on the signal candidates to constrain the particle momenta to point to particular production vertices and constrain the composite particle masses to improve the resolution of the reconstructed objects.

The `nTuples` may also include information about additional particles saved by the selective persistency of the HLT2 and Sprucing lines. This information may be used to create "isolation" information concerning the signal candidate (*i.e.* related to nearby tracks in the event that may be associated with the signal), or for flavour tagging [35]. As the selections in HLT2 and Sprucing are not exhaustive for every possible resonance, one can also use the extra persisted particles to reconstruct excited states for spectroscopy purposes (for example, adding a track to a D^0 candidate to form a D^{*+} or D^{**+}).

The DaVinci application is built on top of Moore and the rest of the LHCb software stack that is used for HLT2 and Sprucing. Therefore, DaVinci shares the same algorithms and tools as HLT and Sprucing, namely the ThOr [11] based selection and combinatorial functions.

At the time of writing, DaVinci writes tuples of `ROOT` columnar TTree objects. In the immediate future, this `nTuple` writing will be updated for thread-safe writing with the Root TTreeWriter. Subsequently, the output type will be changed to the new `RNTuple` [36]. Doing so will bring significant performance enhancements — particularly with regard to memory usage and I/O speed — and significant file size reductions. This is vital to future-proof DaVinci, ensuring its usability as data sample sizes grow rapidly towards LHC Runs 4 and 5.

LHCb Offline Data Processing and Analysis in LHCb Upgrade II

The HL-LHC (LHC Run 4 onwards) is due to begin data collection in 2030. Whereas the general purpose LHC detectors — ATLAS and CMS — will take data at the increased luminosity already in Run 4, LHCb will instead further increase its instantaneous luminosity by a factor five from Run 5 onwards (scheduled for 2036), after the completion of the LHCb Upgrade II. With the resulting instantaneous luminosity of $1.5 \times 10^{34}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$, LHCb aims to achieve a total integrated luminosity of $\approx 300\,\mathrm{fb}^{-1}$ over the lifetime of the HL-LHC.

In Run 5 analysts can naively expect a factor-of-five increase in their `nTuple` data volume. If we maintain current workflows, with analysts still performing significant data processing and filtering steps privately, the `nTuple` data volumes will become unmanageable. Alongside wider file-format and data retrieval R&D, LHCb 's strategy is to move as much of the data processing and filtering that is currently done by analysts into centralised productions run on WLCG resources, achieved by further exploiting the highly

successful Analysis Productions system for more steps in analyses. This is known as the `extended AP model`. In this model Analysis Productions will perform analysis steps including:

- Application of rectangular selection cuts.
- Evaluation of derived quantities with subsequent cuts.
- Machine Learning model application and inference.
- Calibration (*e.g.*, particle identification and tracking efficiencies) routines reducing the number of variables required in final `nTuples`.
- Binning of variables for histogram based analyses.

Generally, these steps are trivially parallelisable and are well suited to grid productions. WLCG resources are experiment agnostic and can run any code/tool that can be packaged to `cvmfs` or deployed via a container. Portable analysis environments thus allow APs to run any application required for the above steps. In particular, the LHCb `lb-conda` tool that provides access to conda environments in `cvmfs` offers a comprehensive (versioned) default environment with all common HEP software packages, as well as the LHCb calibration tools. Alongside this, analysts can create customised environments specifically for their analysis, which are also versioned, enabling full analysis environment preservation.

In moving these steps to APs, analysts must be able to prototype algorithms and cuts on a subset of the data and train any machine learning models. Analysis Productions allow users to request a prescaled dataset, randomly selecting files across a data-taking period, to prototype their analysis. Once finalised, an analysis production can run over the full dataset applying established algorithms, models and cuts. This workflow is well suited to potential future data storage and retrieval methods that reduce the data stored on disk at any one time through performative data recall from tape.

In the current analysis model, the bespoke submission routines and authentications required by the different distributed resources that analysts use frustrates workflows and collaboration. In the `extended AP model`, analysts will leave the WLCG much later in the analysis process, grid submission routines will be handled for the user by the AP system and users will require only a single, familiar authentication step. The resulting derived datasets are registered in LHCb bookkeeping with data provenance ensured and are accessible to all LHCb analysts. The WLCG also benefits from being accessible to the entire LHC virtual organisation, balanced with fair-use policies.

To reduce the data reading load on the WLCG sites, APs will be grouped into so-called “Analysis Trains"whereby physics working groups combine APs running over the same input data, meaning only a single read of the dataset is required. Re-assessing the configuration of the streams of the Spruced data can also alleviate the I/O load. The analysis train model is also well suited to potential future data storage and retrieval methods that opt for performative data recall from tape.

LHCb ’s `extended AP model` is highly dependent on the direction of WLCG over the next decade; LHCb is helping to guide this direction alongside the needs of the ATLAS and CMS collaborations in Run 4.

Acknowledgements The authors express their gratitude to all LHCb collaborators who have enabled the successful development and uptake of the Sprucing and Analysis Productions. In particular we wish to thank the LHCb production managers for running the many, many campaigns, the LHCb data managers for handling our data, and the LHCb Computing project for maintaining and developing the core software upon which we rely.

Author Contributions All authors contributed to this work as part of the DPA project as LHCb The main authors are NS and CB

Funding A.R.W. is supported by UK Research and Innovation under grant`#MR/Y01166X/1.` The work of NG is partially supported by the US National Science Foundation through award PHY-2411665.

Data Availability No datasets were generated or analysed during the current study.

Declarations

Competing Interests The authors declare no competing interests.

References

1. Alves Jr, AA, et al. (2008) The LHCb detector at the LHC. JINST 3, 08005 https://doi.org/10.1088/1748-0221/3/08/S08005
2. Aaij R et al (2024) The LHCb Upgrade. I JINST 19(05):05065. https://doi.org/10.1088/1748-0221/19/05/P05065
3. LHCb Collaboration: Computing Model of the Upgrade LHCb experiment. Technical Report CERN-LHCC-2018-014, LHCb-TDR-018, CERN, Geneva (2018). https://cds.cern.ch/record/2319756
4. Tsaregorodtsev A, Brook N, Casajus Ramo A, Charpentier P, Closier J et al (2010) DIRAC3: The new generation of the LHCb grid software. J Phys: Conf Ser 219:062029. https://doi.org/10.1088/1742-6596/219/6/062029

5. (2020) LHCb Collaboration: RTA and DPA dataflow diagrams for Run 1, Run 2, and the upgraded LHCb detector. Technical Report LHCb-FIGURE-2020-016 . https://cds.cern.ch/record/2730181
6. Frank M (2022) Online RAW Data Format. https://edms.cern.ch/ui/file/784588/1/Online_Raw_Data_Format.pdf, Geneva
7. Collet Y, Kucherawy M (2021) Zstandard Compression and the application/zstd Media Type. RFC Editor . https://doi.org/10.17487/RFC8878 . https://www.rfc-editor.org/info/rfc8878
8. Bird I (2011) Computing for the Large Hadron Collider. Annual Review of Nuclear and Particle Science 61, 99– 118 https://doi.org/10.1146/annurev-nucl-102010-130059
9. LHCb Collaboration: Moore application GitLab repository. https://gitlab.cern.ch/lhcb/Moore
10. LHCb Collaboration: DaVinci application GitLab repository. https://gitlab.cern.ch/lhcb/DaVinci/
11. LHCb Collaboration: ThOr functors. https://lhcbdoc.web.cern.ch/lhcbdoc/moore/master/selection/thor_functors.html
12. Aaij R *et. al.* (2016) Tesla : an application for real-time data analysis in High Energy Physics. Comput. Phys. Commun. **208** arXiv:1604.05596
13. Torvalds L, Hamano J (2010) Git Distributed Version Control System. http://git-scm.com. Accessed: 04-06-2024
14. GitLab Inc.: GitLab. https://about.gitlab.com. Accessed: 04-06-2024 (2011)
15. Grazette L, Hunter R, Noomen E, N, S, Stahl S, Vesterinen M, Zhang S (2025) A Comprehensive Bandwidth Testing Framework for the LHCb Upgrade Trigger System arXiv:2503.19582
16. LHCb collaboration: LHCb computing: Technical Design Report. Technical Report CERN-LHCC-2005-019, CERN, Geneva (2005)
17. Brun R, Rademakers F (1997) ROOT - an object oriented data analysis framework. Nucl Instrum Meth A 389:81–86. https://doi.org/10.1016/S0168-9002(97)00048-X
18. ROOT: JavaScript ROOT. https://github.com/root-project/jsroot
19. LHCb Collaboration: End of successful proton-proton collision data taking period. https://lhcb-outreach.web.cern.ch/2024/10/18/end-of-successful-proton-proton-collision-data-taking-period/
20. Mathe Z (2012) Feicim: A browser and analysis tool for distributed data in particle physics. Technical Report CERN-THESIS-2012-156 http://cds.cern.ch/record/1491175
21. Stewart GA, Mete AS (2024) prmon: process monitor. Zenodo. https://doi.org/10.5281/ZENODO.11400398
22. Burr C, Couturier B, ONeil R, (2024) Facilitating the preservation of LHCb Analyses with APD. EPJ Web of Conf 295:08008. https://doi.org/10.1051/epjconf/202429508008
23. Python community: Python Package Index PyPI. https://pypi.org/
24. Conda-forge community, (2015) The conda-forge Project: community-based software distribution. Zenodo. https://doi.org/10.5281/zenodo.4774216
25. Köster J, Rahmann S (2012) Snakemake–a scalable bioinformatics workflow engine. Bioinformatics 28(19):2520–2522. https://doi.org/10.1093/bioinformatics/bts480
26. LHCb Collaboration: LbCondaWrappers GitLab repository. https://gitlab.cern.ch/lhcb-core/lbcondawrappers
27. Blomer J, Sanchez CA, Buncic P, Charalampidis I, Berzano D (2011) CernVM File System. J Phys: Conf Ser 331(4):042003. https://doi.org/10.1088/1742-6596/331/4/042003
28. Allen C, Dierks T (1999) The TLS Protocol Version 1.0. RFC 2246 . Online; Accessed 23-06-2025
29. Peters AJ, Janyst L (2011) Exabyte scale storage at cern. J Phys: Conf Ser 331(5):052015. https://doi.org/10.1088/1742-6596/331/5/052015
30. Fielding R, Gettys J, Mogul J, Frystyk H, Masinter L, Berners-Lee T, Leach P (1999) Hypertext Transfer Protocol - HTTP/1.1. RFC 2616, Internet Engineering Task Force (IETF). https://doi.org/10.17487/RFC2616
31. XRootD: XRootD project. http://www.xrootd.org/
32. Barrand G, Belyaev I, Binko P, Cattaneo M, Chytracek R, Corti G, Frank M, Gracia G, Harvey J, Herwijnen E, Maley P, Mato P, Probst S, Ranjard F (2001) Gaudi – a software architecture and framework for building hep data processing applications. Comput Phys Commun 140(1):45–55. https://doi.org/10.1016/S0010-4655(01)00254-5. (**CHEP2000**)
33. Mathad A, Ferrillo M, Barré S, Koppenburg P, Owen P, Raven G, Rodrigues E, Serra N (2024) FunTuple: a new n-tuple component for offline data processing at the LHCb experiment. Comput Softw Big Sci 8(1):6. https://doi.org/10.1007/s41781-024-00116-1. arXiv:2310.02433 [physics.data-an]
34. Hulsbergen WD (2005) Decay chain fitting with a Kalman filter. Nucl Instrum Meth A552:566–575. https://doi.org/10.1016/j.nima.2005.06.078
35. Prouve C, Nolte N, Hasse C (2024) Fast Inclusive Flavour Tagging at LHCb arXiv:2404.14145 [hep-ex]
36. Blomer J et al (2024) ROOT's RNTuple I/O Subsystem: The Path to Production. EPJ Web of Conf 295:06020. https://doi.org/10.1051/epjconf/202429506020