Intro to gbasf2

Anil Panta
Justin Guilliams
Jake Bennett

Belle II Summer Workshop
July 13, 2021
Hands-on progress so far:

- Your steering script is prepared
- Using basf2 modules

```python
import basf2
import modularAnalysis as m4
import stdPUs

main_path = basf2.create_path()

m4.inputMdstList(
    environmentType="default",
    fileList=[basf2.FileFile("analysis/tests/mdst.root"],
    path=main_path,
)

list_tree_tuples = list()

# MC Truth
m4.fillParticleListFromMC("pi+ from mc", cut="(dr < 2) and [abs(dz) < 4]", addDaughters=True, path=main_path)
m4.addInclusiveDsterReconstruction(
    "0+charged_0charged_MC -> pi+ from mc",
    slwPlotCut="useCMSFrame(p) < 2",
    DsterCut="useCMSFrame(p) < 2",
    path=main_path)

m4.fillParticleListFromMC("pi0: from mc", "", addDaughters=True, path=main_path)
```
Hands-on progress so far:

- Your steering script is prepared
- Using basf2 modules
- With Input files on local resources

```python
import basf2
import modularAnalysis as ma
import stdPlots

main_path = basf2.create_path()

ma.inputMdsList(
    environmentType="default",
    filelist=[basf2.FileFile("analysis/test/nstd.root")],
    path=main_path,
)

list_tree_tuples = list()

# MC Truth
ma.fillParticleListFromMC("pi+:from_mc", cut="(dr < 2) and (abs(dz) < 4)", addDaughters=True, path=main_path)
ma.addInclusiveDsterReconstruction(
    "0*:charged_smecharged_MC -> pi+:from_mc",
    slowPlotCut="useCMSFrame(p) < 2.0",
    dstarCut="useCMSFrame(p) < 2.0",
    path=main_path)
ma.fillParticleListFromMC("pi0:from_mc", **, addDaughters=True, path=main_path)
```

```
$ basf2 my_analysis_script.py -i /local/path/my_input_files.root
```
Can a whole analysis be performed locally? (on a single computing center or machine)
Belle II:

- Target integrated luminosity: 50 /ab
- Massive data volume ~ hundreds of PetaBytes (PB)
  - Huge storage required
  - Huge computing power required
- Collaborators all over the world
  - Data distributed around the world

Raw and hRaw so far:

- ~7 PB
Distributed ("Grid") Computing:

- Loosely-coupled "super virtual computer"
- Collection of heterogeneous resources for computing, storage, cataloging etc.
Belle II Computing Model:

- One full copy of all raw data is stored at KEK
- Another replica distributed over multiple (6 for now) raw data centers (BNL, DESY, etc.)
- Processed data stored in regional data centers
BelleDIRAC:

- Distributed Infrastructure with Remote Agent Control (DIRAC)
  - acts as a layer between user and resources
  - provides grid solution like: Workload Management (WMS), Account Management, etc.

- BelleDIRAC is an extension of DIRAC
  - It is Belle II specific to fulfill our needs, e.g. production management
Rucio:

- From Jan 2021, Belle II started using Rucio
- Rucio is a Distributed Data Management software
- Rucio acts as a File Catalog (RFC)

File transfer a few hours after the transition
How to interact with the grid?

What tool to use to interact with BelleDIRAC -> Grid?
gbasf2

- gbasf2 is grid-based basf2
- Extends basf2 from local resources to the grid
- Runs the same steering files as for basf2
gbasf2

- gbasf2 is grid-based basf2
- Extends basf2 from local resources to the grid
- Runs the same steering files as for basf2

$gbasf2$

OR

$gb2\_tools$

gb2_tools

- gb2_tools: set of BelleDIRAC cmd line tools
- Used to monitor, manipulate jobs on the grid
- Additional functionality for grid management
gbasf2 workflow:

- Create a basf2 steering file
- Run basf2 locally and make sure it works
- Find the input files on the grid
- Submit a gbasf2 project

Steering file

```python
[Import statements]

main = basf2.create_path()

# Find input files
list_files = list()

# Process input files
for file in list_files:
    process_file(file)
```
How to use gbasf2?

- Fulfill the prerequisites:
  1. A valid grid certificate issued within a year and installed in ~/.globus and in a web browser
  2. Belle VO membership registered or renewed within a year: https://voms.cc.kek.jp:8443/voms/belle
  3. DIRAC registration: https://dirac.cc.kek.jp:8443/DIRAC/
- Install gbasf2 or use pre-installed version in CVMFS. gbasf2install

$ gbasf2 <steering_scripy.py> -i <input_path> -p <project_name> -s <basf2_release>

- project_name: name assigned by you
- basf2_release: any available Basf2 software version
- input_path: the logical file name (LFN) of the files to analyze

Detail in hands on by Justin.
Input files:

- In basf2 where input files are local:
  
  $ basf2 my_analysis_script.py -i /local/input/files.root

- In gbasf2 where input files are in storage element in grid:
  
  $ gbasf2 my_analysis_script.py -i <input_files> -s <basf2_release> -p <project_name>

<input_files>
WHERE??
WHAT??
The physical path name (PFN) of the file is the actual location of files on grid. For example,
srm://dcache.ijs.si:8443/srm/managerv2?SFN=/pnfs/ijs.si/belle/DATA/belle/MC/release-05-02-03/DB00001363/SkimM14ri_ax1/prod00017415/e1003/4S/r00000/mixed/17241100/udst/sub00/udst_000068_prod00017415_task10020000069.root

This looks complicated. And identifying a PFN for every file is cumbersome and not appropriate for gbasf2 use.
Input files location ‘WHERE’?

- The physical path name (PFN) of the file is the actual location of files on grid. For example,
  
  srm://dcache.ijs.si:8443/srm/managerv2?SFN=/pnfs/ijs.si/belle/DATA/belle/MC/release-05-02-03/DB00001363/SkimM14ri_ax1/prod00017415/e1003/4S/r0000/mixed/17241100/udst/sub00/udst_000068_prod00017415_task1002000069.root

- This looks complicated. And identifying a PFN for every file is cumbersome and not appropriate for gbasf2 use.

- To keep track of the location of files and replicas, we use a File Catalog (FC), specifically the Rucio FC

- Abstraction of PFN -> Logical File Name (LFN), looks like:

  /belle/MC/release-05-02-03/DB00001363/SkimM14ri_ax1/prod00017415/e1003/4S/r0000/mixed/17241100/udst/sub00/udst_000068_prod00017415_task1002000069.root

- The above path is enough to locate files and replicas on the grid

- Each path is unique
Terminology:

- Units of data management:
  - **File**: LFN (Logical File Name), smallest unit of data
    
    `/belle/MC/release-05-02-03/DB00001363/SkimM14ri_ax1/prod00017415/e1003/4S/r00000/mixed/17241100/udst/sub00/udst_prod00017415_task10020000069.root`
Terminology:

- **Units of data management:**
  - File: LFN (Logical File Name), smallest unit of data
    
    /belle/MC/release-05-02-03/DB00001363/SkimM14ri_ax1/prod00017415/e1003/4S/r00000/mixed/17241100/udst/sub00/udst_prod00017415_task10020000069.root
  - **Datablock**: LPN (Logical Path Name), a block of at most 1000 files
    
    /belle/MC/release-05-02-03/DB00001363/SkimM14ri_ax1/prod00017415/e1003/4S/r00000/mixed/17241100/udst/sub00
Terminology:

- Units of data management:
  - File: LFN (Logical File Name), smallest unit of data
    ![File example]
  - Datablock: LPN (Logical Path Name), a block of at most 1000 files
    ![Datablock example]
  - Dataset: LPN (Logical Path Name), consists of at least 1 dataset
    ![Dataset example]
MetaWHAT?

- What kind of files do the LFNs represent?
  -> Metadata, details or description of files/datablocks
- Each LFN and LPN has metadata associated with it
- All metadata are described here

Metadata of dataset:

creationDate: 2021-06-15 14:23:19
lastUpdate: 2021-06-19 14:51:17
nFiles: 1
size: 1065076873
status: good
productionId: 18887
transformationId: 525947
owner: g:belle_dataprod
mc: proc12
stream: data
dataType: mdst
dataLevel: mdst
beamEnergy: 45
mcEventTypeName: hadron
skimName: release-05-01-22
dbGlobalTag: DB00081779
sourceCode: steeringFile: data/data_processing/rec/runRec.py
steeringFileRevision: 12
experimentIdLow: 12
experimentIdHigh: 12
runIdLow: 5351
runIdHigh: 5351
logFm:
parentDatasets:
description: Exp 0012 - proc12 - hadron - run range: 4814-5474
Dataset-Searcher (DSS)

- The DSS is the **tool to find datasets on the grid via their associated metadata**
  - Users select/input relevant metadata of interest
  - Returns a list of datasets matching the metadata
  - Use the dataset as input to gbasf2

- Two ways to use the DSS
  1. **gb2 tool**: `gb2_ds_search`

* Details in the hands-on session by Justin
Let's move to the hands-on session for the DSS and gbasf2