DST file & dstparser

Kozo Fujisue

Academia Sinica

(~~TAML 2024~~ workshop @ Academia Sinica)

TADAML 2025

What is DST?

DST: **D**ata **S**torage **T**ape

- A binary-file format for the TA experiment
  - DST is originally HiRes format (the data was stored on magnetic tape)
  - Observational & simulated data of SD & FD in TA (TALE(-infill), TA×4) are stored in this format
    - and converted into a certain format afterwards depending on analysis
    - The same reconstruction/analysis codes can be applied to Obs./MC data

- There are some software in TA to parse/analyze the DST files
  - sdanalysis
  - TA JAVA
  - …
  - and **dstparser**, newly developed in Taiwan
    - dstparser uses sdanalysis

https://github.com/TA-DNN/dstparser

SD Raw data

Raw data (in ASCII format)

1. Communication tower collects information from each SD.
   - Waveforms (when a trigger is issued)
   - SD status:
     trigger rate, communication err, etc.

2. The collected data are transferred to servers CRC (a facility of TA near the TA site), and servers of institutes.

3. The raw data in ASCII format are converted to DST files in serves of institutes.

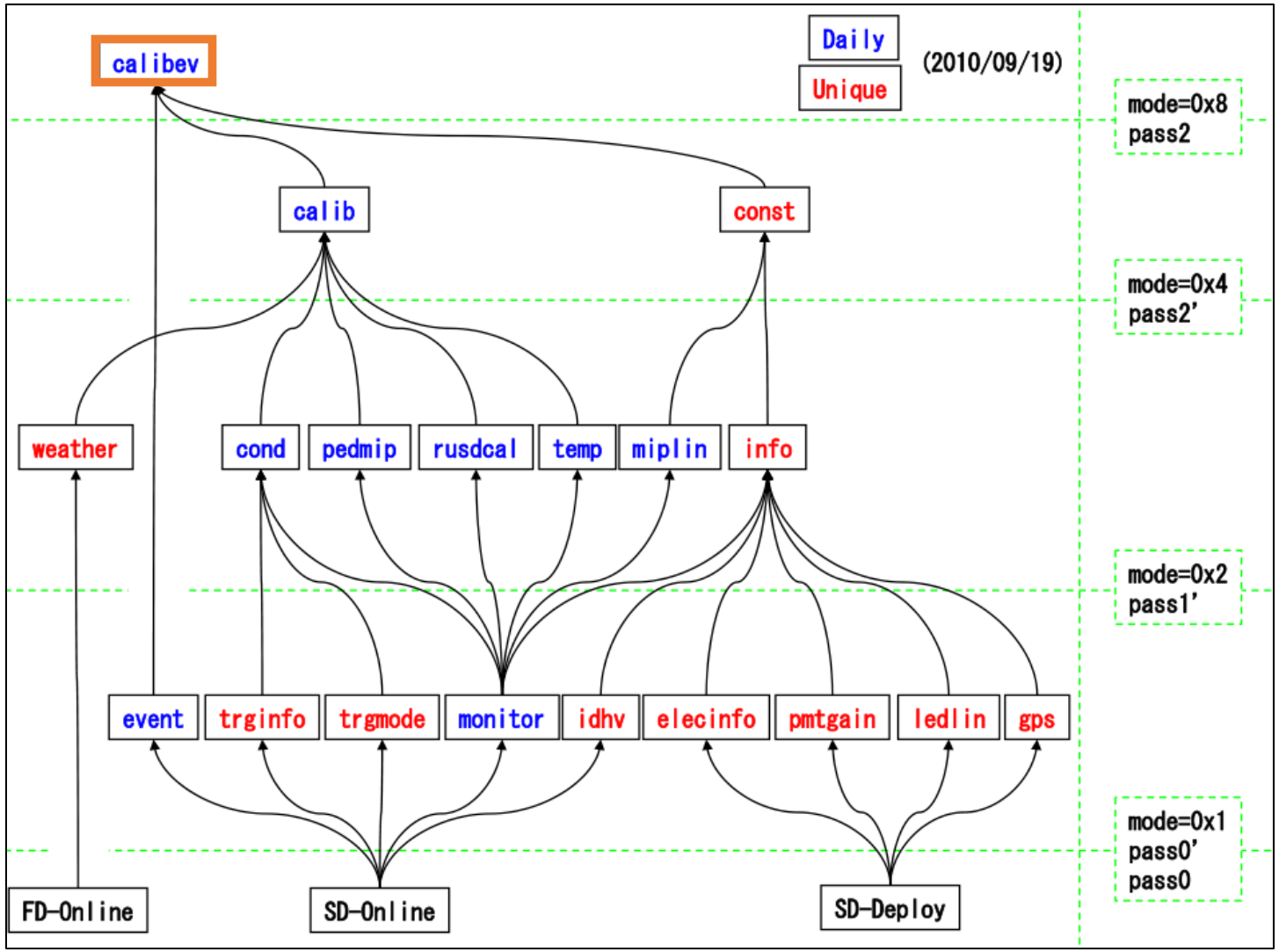
SD monitor data

```
#T 00007060 241017 031513 778 313 0
#A 0 0 0 0
#C 0 0 0 0
#e2 0 1379596e
#t 036333
#t 060206
L 4716 fffd 0 313 0 021 2faedc4 a3 90 9b 83
L 4815 ffff 0 313 0 015 2faee0b 8d 91 83 8d
L 4419 ffff 0 313 0 021 2faeb8e 97 ad b0 9d
L 4514 ffff 0 313 0 024 2faee6c 73 61 7e 71
###ERR L 4814 000 0 Dead
L 4412 ffff 0 313 0 012 2faedab 63 5d 55 75
L 4515 ffff 0 313 0 014 2faeab2 7f a3 94 96
L 4408 ffff 0 313 0 020 2faeddb 8a 77 98 8a
L 4713 ffff 0 313 0 023 2faedff c5 98 a8 88
L 4617 ffff 0 313 0 023 2faec1e a4 b6 88 99
L 4614 ffff 0 313 0 012 2faeea4 85 88 7f 73
L 4513 ffff 0 313 0 016 2fae9b9 6c 81 75 61
L 4613 ffff 0 313 1 026 2faeb1c 8c 83 7b 8c
L 4710 ffff 0 313 0 022 2faee18 9a 97 83 80
```

Waveform data

```
#t 426685 ffffffff 1 781
#t 426880 3 0 0 0
E 00000000 1379596e
W 4613 1
w 0 0 132
41d370b1
037e028b
00006005
01007005
02007004
03007005
04006005
05006005
06007005
07007005
08007005
09007005
0a007005
0b008005
0c006004
0d006004
0e007005
```

DST file (of observational SD data)



time-dependent info

time-independent info

Some steps of conversion:

- Many DST files are generated
  - DST files has ".dst" extension
  - Some C++ structures:
    (e.g.) calibev: calibrated event data
    rufldf: LDF fit results

- The last one, calibev , has all information needed for analysis
  - Air shower event data
    - waveforms, timing, …
  - Calibration data
    - MIP peak, pedestal, …
  - calibev file is generated per one day
    (e.g.) tasdcalibev_pass2_220131.dst
    has data on 31st Jan. 2022

- (c.f.) calib & const data is used in MC simulation

DST file (for SD)

sdanalysis_2019/dst2k-ta/inc/
tasdcalibev_dst.h

```
typedef struct {
  short site;    /* site id */
  short lid;     /* position id */

  int clock;     /* clock count at the trigger timing    */
  int maxClock; /* maximum clock count between 1PPS      */
  char wfId;     /* waveform id in the trigger           */
  char numTrgwf;/* number of triggered waveforms */
  char trgCode; /* level-1 trigger code                  */
  char wfError; /* broken waveform data or saturate
                   LSB   : upper PMT saturate (with LED data)
                   bit-1 : lower PMT saturate (with LED data)
                   bit-2 : upper PMT saturate (with MIP data)
                   bit-3 : lower PMT saturate (with MIP data)
                   bit-4 : upper FADC saturate
                   bit-5 : lower FADC saturate
                   bit-6 : daq error (FPGA to SDRAM) */
  short uwf[tasdcalibev_nfadc]; /* waveform of the upper layer */
  short lwf[tasdcalibev_nfadc]; /* waveform of the lower layer */

  float clockError; /* fluctuation of maximum clock count [ns] */
  float upedAvr;    /* average of pedestal (upper) */
  float lpedAvr;    /* average of pedestal (lower) */
  float upedStdev;  /* standard deviation of pedestal (upper) */
  float lpedStdev;  /* standard deviation of pedestal (lower) */
```

calibev: calibrated event data

sdanalysis_2019/dst2k-ta/inc/
rufldf_dst.h

```
typedef struct
{
  /* [0] - LDF alone fit, [1] - LDF + geom. fit */
  real8 xcore[2];  // Core position + uncertainty
  real8 dxcore[2];
  real8 ycore[2];
  real8 dycore[2];
  real8 sc[2];     // Scaling factor in fron of LDF + Minuit uncertainty on that
  real8 dsc[2];
  real8 s600[2];   // S600, VEM/m^2
  real8 s600_0[2];
  real8 s800[2];   // S800, VEM/m^2
  real8 s800_0[2];
  real8 aenergy[2]; /* Energy in EeV using AGASA formula */
  real8 energy[2];  /* Energy in EeV using Rutgers formula */
  real8 atmcor[2];  /* Energy atmopsheric correction factor that was applied */
  real8 chi2[2];

  /* These variables are from combined LDF/geom fit */
  real8 theta;
  real8 dtheta;
  real8 phi;
  real8 dphi;
  real8 t0;
  real8 dt0;
```

rufldf: LDF fit results

Parse DST file with sdanalysis

$ source sdanalysis_env.sh
$ **dstdump.run** DAT020020_gea.dat.hrspctr.1945.rufldf.dst.gz

(a DST file which contain some MC-simulated events of ~100 EeV proton)

```
START OF EVENT ************************************************************
rusdmc :
Event Number: 900418644
Corsika Particle ID: 14
Total Energy of Primary Particle: 104.002 EeV
Height of First Interaction: 39.9204 km
Zenith Angle of Primary Particle Direction: 55.3925 Degrees
Azimuth Angle of Primary Particle Direction: 127.496 Degrees (N of E)
Counter ID Number for Counter Closest to Core: 1917
Position of the core in CLF reference frame: (8649.38,100.55,33.1277) m
Time of shower front passing through core position: 40210760 x 20 nsec
rusdraw :
event_num 900418644 event_code 0 site BRLRSK run_id: BR=111 LR=111 SK=111 tr
ig_id: BR=1212 LR=1212 SK=1212
errcode 0 date 05/23/09 19:51:52.804215 nofwf 64 monyymmdd 090523 monhhmmss
195152
wf# wf_id  X   Y    clkcnt     mclkcnt   fadcti(lower,upper)  fadcav      pc
hmip       pchped     nfadcpermip     mftchi2      mftndof
00    00    1  10   40210263   50000000      648      645    37   37     98
     98    37    37     42.1   42.7   21.0   31.2     24   24
lower fadc
```

Event input information

```
lower fadc
     5     4     5     5     5     5     6     3     4     4     4
     4
     4     4     4     5     5     4     5     3     4     4     6
     5
     6     3     3    10    15    18    16    12     9     7     5
     5
     5     3     3     5     6     4     5     6     4     5     3
     5
     6     5     5     6     4     5     4     4     5     5
     5
     3     4     4     6     5     5     4     4     6     7     4
     5
     4     6     4     5     4     5     5     5     5     5     5
     5
     6     6     5     6     6     4     5     6     4     5     4
     4
     5     4     3     3     7     4     4     4     5     5     4
     3
     4     5     5     5     5     6     4     4     4     4     5
     5
     5     4     6     3     6     5     4     4
upper fadc
     5     6     5     5     4     5     6     4     5     6     4
     6
     5     4     4     4     3     3     6     4     4     5     6
```

Waveform information
(FADC count for 128 bins)

```
rufldf :
xcore[0] 3.77 dxcore[0] 0.07 ycore[0] 18.33 dycore[0] 0.03 s800[0] 69.69 energy[0] 11
0.84 atmcor[0]: 1.00 chi2[0] 6.92 ndof[0] 11
xcore[1] 3.85 dxcore[1] 0.07 ycore[1] 18.34 dycore[1] 0.03 s800[1] 69.55 energy[1] 88
.86 atmcor[1]: 1.00 chi2[1] 20.42 ndof[1] 22
theta 52.66 dtheta 0.88 phi 22.84 dphi 0.86 t0 8.78 dt0 0.05
bdist 0.49 tdist 0.56
```

Reconstruction information

Triggered SD information
(pedestal level, 1mip peak, …)

Parse DST file with sdanalysis

$ source sdanalysis_env.sh
$ **sditerator**_add_standard_recon.run DAT020020_gea.dat.hrspctr.1945.rufldf.dst.gz

Event information

```
#EVENT DATA
14 1.052238e+02 0.966782 2.041773 -101813.664062 597053.875000 3612.088623 8
0602 94056 26 62 213417 97.153997 31469.648899 2201.505632 22.167569 20 11.5
29052 0.043776 21.190034 0.041083 61.318981 54.610267 118.060786 0.578170 0.
707776 8.472103 21 8.897328 0.118648 3.622117 1.274462
#SD meta DATA
0110 01 2.418326 2.418326 10.539463 7.996258 -11.270333 -6.405030 0.074293 3
8.936992 47.334641 0115 01 9.613345 9.613345 1.470921 0.818203 -11.247216 -1
.384837 0.053929 34.502202 38.499012 0206 01 4.631793 4.631793 2.479228 0.87
8569 -10.196882 -10.371078 0.047891 38.772152 42.967603 0206 01 5.171420 5.1
71420 1.424992 2.231332 -10.196882 -10.371078 0.047891 38.772152 42.967603 0
514 01 0.634561 0.634561 1.290165 1.620493 -7.248980 -2.428786 0.022258 41.5
64464 37.025770 0516 01 10.957414 10.957414 1.138173 1.299749 -7.246043 -0.4
28118 0.030335 39.976350 37.507235 0517 01 9.353525 9.353525 1.824497 0.6537
79 -7.241344 0.569441 0.035636 38.229709 41.298354 0519 01 7.160043 7.160043
 1.246047 1.252226 -7.252092 2.561823 0.056441 39.324351 39.928906 0612 01 1
2.736183 12.736183 0.930627 1.238012 -6.249659 -4.433266 0.026411 37.609067
36.954408 0722 01 8.833884 8.833884 1.430481 1.375387 -5.233613 5.573915 0.0
73573 37.574778 40.897568 0820 02 1.913675 1.913675 5.282276 10.052153 -4.23
9170 3.569672 0.031727 41.459404 38.399733 0908 01 10.952418 10.952418 2.395
963 1.517915 -3.239048 -8.450594 -0.008320 44.241082 39.857292 0912 01 1.888
```

SD information

```
#SD waveform DATA
0110 10669584 50000000 7 8 8 9 8 7 7 8 5 8 8 7 7 8 8 6 6 7 8 7 8 8 7 8 8 8 7
 7 8 44 91 106 89 63 41 23 12 10 8 7 6 8 8 9 9 8 7 8 7 7 8 8 7 7 7 9 8 7 7 8
 7 7 9 7 6 9 7 8 8 6 7 7 8 8 9 8 8 8 8 8 8 7 7 8 8 7 8 7 9 8 7 8 7 7 7 7 8 9
 8 8 8 7 8 7 6 7 7 6 8 8 9 7 9 8 7 8 8 8 7 6 8 8 7 8 7 7 8 7 7 6 7 6 7 7 8 7
 7 6 8 7 7 6 6 6 6 6 7 8 7 8 7 6 7 5 6 8 8 42 84 96 81 57 36 22 14 9 7 7 9 6
 7 7 7 5 7 7 7 7 7 7 8 6 6 8 7 7 8 6 8 6 7 7 7 6 6 7 7 6 6 6 7 6 8 7 5 7 8 7
 6 7 7 6 7 8 7 7 6 7 8 6 6 6 7 8 8 7 7 8 8 8 8 7 7 7 8 7 8 8 7 7 7 6 5 6 7 6
 6 7 7 8 6 7 9 8 6 7 0115 10671026 50000000 5 5 4 6 5 4 5 5 5 4 4 6 5 4 6 6
5 5 5 4 3 3 4 4 4 4 6 9 15 16 16 12 9 7 6 5 6 5 7 5 5 5 4 5 5 5 4 5 5 6 3 5
4 5 5 4 4 4 6 5 5 5 5 6 5 5 5 5 5 6 4 5 5 4 5 4 4 3 5 6 5 6 6 4 5 5 5 4 6 4
5 5 5 4 4 4 4 5 5 5 5 5 5 2 6 5 6 6 5 5 3 6 6 6 4 5 7 5 5 7 5 4 5 4 4 5 5 5
8 6 5 5 7 6 7 5 7 6 5 6 8 6 6 7 7 6 8 5 5 6 6 6 7 7 6 11 12 14 13 11 8 6 7 7
 7 5 7 6 6 6 8 7 5 5 7 6 6 7 7 5 5 7 7 5 7 6 7 7 6 5 6 6 6 5 5 6 6 7 6 6 5 6
 7 5 6 6 6 6 7 5 8 6 7 5 4 7 6 7 6 7 6 8 7 5 6 7 7 7 6 6 7 6 6 5 7 6 5 6 6 6
```

Waveform information

(installation of sdnalaysis_2019 is described in backup,
only for TA collaborators)

Parse DST file with dstparser

https://github.com/TA-DNN/dstparser

Event information

```
#EVENT DATA
14 1.052238e+02 0.966782 2.041773 -101813.664062 597053.875000 3612.088623 8
0602 94056 26 62 213417 97.153997 31469.648899 2201.505632 22.167569 20 11.5
29052 0.043776 21.190034 0.041083 61.318981 54.610267 118.060786 0.578170 0.
707776 8.472103 21 8.897328 0.118648 3.622117 1.274462
#SD meta DATA
0110 01 2.418326 2.418326 10.539463 7.996258 -11.270333 -6.405030 0.074293 3
8.936992 47.334641 0115 01 9.613345 9.613345 1.470921 0.818203 -11.247216 -1
.384837 0.053929 34.502202 38.499012 0206 01 4.631793 4.631793 2.479228 0.87
8569 -10.196882 -10.371078 0.047891 38.772152 42.967603 0206 01 5.171420 5.1
71420 1.424992 2.231332 -10.196882 -10.371078 0.047891 38.772152 42.967603 0
514 01 0.634561 0.634561 1.290165 1.620493 -7.248980 -2.428786 0.022258 41.5
64464 37.025770 0516 01 10.957414 10.957414 1.138173 1.299749 -7.246043 -0.4
28118 0.030335 39.976350 37.507235 0517 01 9.353525 9.353525 1.824497 0.6537
79 -7.241344 0.569441 0.035636 38.229709 41.298354 0519 01 7.160043 7.160043
 1.246047 1.252226 -7.252092 2.561823 0.056441 39.324351 39.928906 0612 01 1
2.736183 12.736183 0.930627 1.238012 -6.249659 -4.433266 0.026411 37.609067
36.954408 0722 01 8.833884 8.833884 1.430481 1.375387 -5.233613 5.573915 0.0
73573 37.574778 40.897568 0820 02 1.913675 1.913675 5.282276 10.052153 -4.23
9170 3.569672 0.031727 41.459404 38.399733 0908 01 10.952418 10.952418 2.395
963 1.517915 -3.239048 -8.450594 -0.008320 44.241082 39.857292 0912 01 1.888
```

SD information

dstparser/src/dstparser/**dst_adapter.py**

```python
def standard_recon(data, dst_lists):
    # Date of event
    # rusdraw_.yymmdd = 80916; // Event date year = 08, month = 09, day = 16
    data["std_recon_yymmdd"] = event_list[7]
    # Time of event
    # rusdraw_.hhmmss = 1354;  // Event time, hour=00, minute=13, second = 54
    data["std_recon_hhmmss"] = event_list[8]
    # Microseconds for the second
    # rusdraw_.usec = 111111
    data["std_recon_usec"] = event_list[11]
    # Number of waveforms for event for all detectors (not very useful so far)
    # data["std_recon_nofwf"] = event_list[10]
    # number of SDs in space-time cluster
    data["std_recon_nsd"] = event_list[9]
    # energy reconstructed by the standard energy estimation table [EeV]
    data["std_recon_energy"] = event_list[12]
    # reconstructed scale of the Lateral Distribution Function (LDF) fit [VEM m-2]
```

1) Reads DST files using sditerator
2) Makes python dictionary, which contains event info, SD info, and waveform info.
3) Generates HDF5 files.
   The srecog uses the generated HDF5 files

Summary

- DST (Data Storage Tape) is a unique file format for TA (SD & FD)
  - The tasdcalibev files include event & calibration data of SDs
- There are some softwares to parse DST files
  - sdanalysis is a package for SD-reconstruction & -analysis
- A new python package was developed: dstparser
  - It reads DST files using sdanalysis, and makes HDF5 files
    - The generated HDF5 files are used for SD-ML.
  - To use dstparser, followings are required:
    - sdanalysis (& ROOT v5.34)
    - Python3 (with some libraries)

Backup

How to install/use sdnalaysis_2019

1. Install and compile sdnalaysis_2019 package
   - see Japanese TA Wiki (for TA collaborators only)

2. Install ROOT v5.34 (if not available in your environment)
   For example:
   $ git clone https://github.com/root-project/root.git
   $ cd root
   $ git checkout -b v5-34-36 v5-34-36
   $ mkdir bld; cd bld
   $ make -j4
   $ make install

3. Edit sdnalaysis_2019/sdanalysis_env.sh
   and modify root path: source /…/root/bin/thisroot.sh

4. $ source sdnalaysis_2019/sdanalysis_env.sh
   then you can use the sdnalaysis_2019 package

sditerator

The source codes of sditerator:
sdnalaysis_2019/sditerator/src/

How to edit/compile sditerator (in your environment):

1. $ source sdanalysis_2019/sdanalysis_env.sh
2. Edit sdnalaysis_2019/sditerator/src/**sditerator_cppanalysis.cpp**

   The output format is written in the cpp file.
3. Compile:

   $ cd sdnalaysis_2019/sditerator

   $ ./rebuild.sh

   sdnalaysis_2019/bin/sditeerator.run will be built